UNIVERSIDAD CARLOS III DE MADRID
ESCUELA POLITÉCNICA SUPERIOR
Departamento de ingeniería eléctrica

INGENIERIA DE TELECOMUNICACIONES

PROYECTO FIN DE CARRERA

ANÁLISIS TIEMPO-FRECUENCIA DE LA SEÑAL DE VIBRACIÓN DE UN OLTC (CAMBIADOR DE TOMAS EN CARGA) APLICANDO WAVELET DISCRETA

Autor: José María Lucas Zaragoza
Tutor: Edwin Rivas Trujillo
Codirector: Juan Carlos Burgos *Díaz*

AGRADECIMIENTOS

En primer lugar expresar mis agradecimientos a mis tutores, Edwin Rivas y Juan Carlos Burgos, por su gran ayuda.

A mis Padres, José María y Ana María, por disfrutar de mis méritos y sufrir con mis fracasos, su apoyo y su ayuda constante, gracias por que sin ellos nada habría sido posible.

A mi hermana Ana por soportarme, y a mi hermano Rubén porque me aporta fuerzas, alegría y las ganas de luchar para crecer y poder ayudarle en todo lo que pueda.

A mi novia Gema, por el tiempo alejados y aún así tan cerca, el tiempo que no he podido estar con ella en las noches de estudio, por saber esperar, ser la mejor compañera de este duro viaje, por saber mis defectos y aún así quererme.

A mis amigos y amigotes, como Chocano y Valverde, que tanto ayudan a desconectar y que me permiten disfrutar y ver la vida con otros ojos. Me he perdido muchas pero he disfrutado de más.
También a mis amigos durante la carrera, de residencia, de lugares y fiestas, de exámenes y prácticas, de convivencia (viva Eresma) idas y venidas, gracias Nito, Pedro y Vilo…
Tener un amigo... es una gracia. Conservar un amigo... es una virtud. SER VUESTRO AMIGO ES UN HONOR.

A mi gran familia, a mis tíos, tías y a esos primazos, como Santi y Carlos e Inés, que es un placer juntarme con ellos y que así siga por muchos años.

A mis abuelas. Teresa y Pilar, porque son únicas, y sobre todo a mis dos abuelos, Chico y Juan, porque desde la distancia me llega su ayuda y su fuerza, y se les recuerda con mucho cariño.

A todos los que no nombro, pero que sabéis lo importante que habéis sido y sois.

Siempre os estará muy agradecido y en mí encontrareis apoyo.

INDICE

INDICE DE ILUSTRACIONES

INDICE DE TABLAS

CAPITULO1. INTRODUCCIÓN

1.1. JUSTIFICACION

(Ref. [5]) El proyecto está enmarcado dentro del ámbito de desarrollo de nuevas técnicas para contribuir a la evolución del mantenimiento predictivo, más concretamente en la diagnosis de averías mediante el análisis de la medición, usando acelerómetros piezoeléctricos de la señal de vibración de una operación del CTC, para su comparación con una señal patrón.

Particularmente se trata del cambiador de tomas en carga (CTC), el cual es la única parte con movimiento en un transformador eléctrico, siendo la responsables del 40 por ciento de las averías de estas máquinas. Su función es realizar la operación de cambio de una toma del arrollamiento de regulación a otra, permitiendo regular la tensión de salida del transformador a los niveles requeridos sin la interrupción de la corriente de carga. La valoración de su estado electromecánico es importante para asegurar la fiabilidad del transformador y, por tanto, del sistema de transmisión de energía eléctrica al cual se encuentre conectado, evitando posibles apagones que repercutan en el bienestar de la sociedad en aspectos como seguridad ciudadana, sanidad, transporte, etc.

Como en casi cualquier proceso, el fin principal es el ahorro de tiempo y dinero, el cuál vendrá dado por el seguimiento de la evolución del proceso industrial con el fin de detectar los posibles fallos y sus causas, a fin de evitarlos y conseguir una operatividad eficiente.

En el caso planteado el análisis se hace aún más difícil, ya que las vibraciones de un sistema tan complejo como el CTC no son exactamente repetibles al realizar la misma maniobra dos veces, con lo que se debe procesar la señal e implementar indicadores de diagnostico.

Por tanto se hace necesario la implementación de una herramienta de análisis, que permita, de una manera rápida, cómoda y eficaz , obtener los indicadores de diagnostico necesarios para clasificar cada una de las señales con un determinado tipo de avería.

1.2. MARCO GLOBAL DEL PROYECTO

Como meta general del mismo, se tratará de desarrollar una técnica de diagnóstico de averías a través de la vibración de un elemento de los transformadores eléctrico de potencia.

Idealmente una máquina no produce variaciones, dado que toda la energía se debería emplear en el trabajo que debe realizar, luego en el caso planteado, se produce una disipación de energía en forma de vibración.

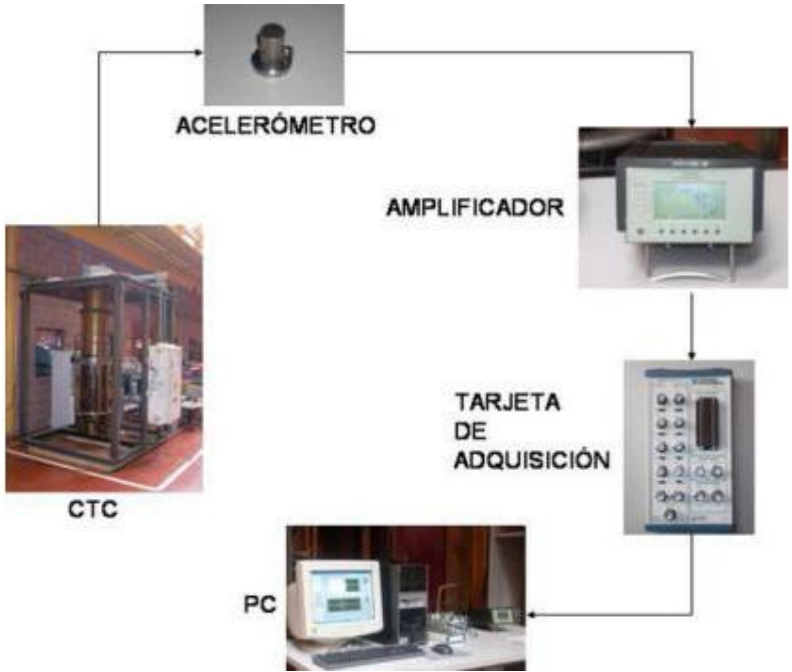
A fin de evitar un mantenimiento correctivo, reemplazando los componentes de las máquinas una vez se averían, dando lugar a pérdidas de dinero y tiempo, y supuestos daños colaterales de la misma, se podría plantear un mantenimiento preventivo. Este último tampoco es la mejor opción, dado que se podría evitar cambiar las piezas que actualmente tienen una vida útil. (Ref [5])

Por tanto, el objetivo en el cuál se enmarca el desarrollo de la herramienta de análisis, es hacer un mantenimiento predictivo, el cual se basa en el estado actual de la máquina, necesitando para ello una diagnosis óptima de la misma, basándose e en la señal de vibración procedente de un intercambiador de tomas en carga.

Las variables a manejar son principalmente: número de impulsos, tiempo entre ellos, contenido de energía del impulso y banda de frecuencia asociada al mismo.

Como proceso previo a la obtención de la señal descrita, se emplean acelerómetros para captar las vibraciones de la máquina, de esta forma se registrar las señales, que pasan por un amplificador y una tarjeta de adquisición de datos antes de llegar al ordenador.

Con el objetivo de extraer la información, se hace necesario procesarlas y obtener una



serie de indicadores de diagnostico. Parte del proceso se observa en la ilustración1.

Para cualquier diagnóstico se necesita una señal en el dominio del tiempo variable en amplitud, sin embargo la información que se puede obtener con esta representación no tiene por qué ser la más apropiada, ya que la información que caracteriza la señal puede observarse más claramente en el dominio de la frecuencia, mediante un espectro que muestre las frecuencias existentes en la señal. Es por ello que se exige una representación en ambos dominios.

Para obtener la información en frecuencia de la señal existen muchos tipos de transformadas, de las cuales, la transformada de Fourier es sin duda la más conocida y utilizada. Sin embargo se dan ciertos problemas al usar la transformada de Fourier con señales no estacionarias, ya que es incapaz de proporcionar información temporal de las frecuencias existentes, pudiendo conocer todas las frecuencias que aparecen en la señal, pero no identificar cuando se producen.

La Short Time Fourier Transform (STFT) ,mostrada en la ilustración2, resuelve estas limitaciones, sin embargo existe un problema cuyo origen es el principio de incertidumbre de Heisenberg; no siendo posible la representación exacta tiempo-frecuencia de una señal, sino tan sólo los intervalos de tiempo en los cuales existen determinadas bandas de frecuencia,

Ilustración 1 . Adquisición de la señal (Tomada de la referencia [5])

apareciendo un problema de resolución.

Ilustración 2. Short time Fourier transform

(Ref.[3]) Este problema de resolución tiempo frecuencia aparece con cualquier tipo de transformada, sin embargo la transformada de wavelet ofrece la posibilidad de estudiar la señal en diferentes escalas y resoluciones.

La clave de la transformada reside, en que esta proporciona una buena resolución temporal y pobre resolución en frecuencia para altas frecuencias y buena resolución en frecuencias y baja en tiempo para frecuencias bajas. Este hecho adquiere vital importancia en el caso que nos atañe, dado que la señal a tratar tiene altas frecuencias de corta duración y componentes de baja frecuencia en larga duración (típico en señales de vibración).

A partir de aquí, se debe extraer las características de la señal capaces de captar la verdadera información acerca de las averías, las cuales se encontraran en las correspondientes componentes locales de la señal.

Dicha transformada de wavelet, permite por tanto diferentes niveles de localización en el dominio del tiempo y de la frecuencia. Lo cual permite prever un correcto diagnostico a través de la información almacenada en una señal no estacionaria.

El uso por tanto de un software matemático se hace imprescindible a la hora de evaluar las características e información de la señal con la que se debe tratar. El software seleccionado para dicho propósito es Matlab 7.0. , dada la gran facilidad de uso en entornos de cálculo matemático y de ingeniería en su sentido amplio, también cabe destacar la gran utilidad de las herramientas que existen dentro del programa para trabajar con wavelet, que ayudan en gran medida al desarrollo del trabajo.

Como se ha comentado anteriormente, una de las complicaciones del desarrollo del análisis con la transformada de wavelet discreta, al estar en fase experimental, están escritas desde el punto de vista teórico matemático con lo que su orientación práctica resulta bastante laboriosa.

Dicho esto, la herramienta de análisis permitirá, en su desarrollo global, el estudio, seguimiento, y diagnostico de averías, de una señal de vibración, como es la de un cambiador de tomas de carga y una gran ayuda para el análisis de señales no estacionarias, usando la transformada de wavelet en cualquier software, pero más particularmente en Matlab. (Todo el tratamiento e implementación será incluido en el anexo, de una manera clara y explicativa, a modo que sirva también de guía para futuros trabajos y estudios en este campo).

1.3. OBJETIVOS DEL PROYECTO

Una vez enmarcado el proyecto, el objetivo es el desarrollo de una herramienta que permita analizar la señal de vibración procedente del cambiador de tomas en carga (señales que ya se tienen preparadas para su análisis, para la extracción de las características de los impulsos que existen en ella.)

Para el procesado de la señal y la obtención final de estos coeficientes se establecen los siguientes objetivos intermedios:

1. Elaborar una interfaz grafica, la cual permita la interactuación entre el usuario y la herramienta de análisis desarrollada para permitir la clasificación de la señal entre señal en buen estado o señal con avería.
2. Para la implementación de esta herramienta, se desarrolla un primer método (cuya base teórica se encuentra en el capítulo 3), el cual será plasmado mediante código en Matlab (análisis de tipo matricial).
   Esta implementación es independiente de las funciones wavelet de Matlab, y permitirá el análisis multirresolución (división en frecuencia) de la señal mediante la elección de diferentes Wavelet madre.
3. Tratamiento y compresión de la información de la señal de entrada, compuesta, en principio, por 500000 muestras, para poder ser tratada posteriormente, de forma que se mantenga la información que se quiere extraer, independientemente de la señal de entrada.
4. Obtención de las principales características de los impulsos de la señal, (mediante la transformada de wavelet, y hacer uso de las funciones de Matlab relacionadas con los distintos tipos de wavelet madres, para , en primer lugar,

establecer otra posible vía de análisis, y , en segundo lugar, establecer una comparación entre ambos métodos que demuestre su fiabilidad.

5. Obtención de coeficiente de la transformada de wavelet madre, con el fin de recomponer la señal.

Las características que se pretenden extraer son:

- Número de impulsos de vibración durante la conmutación.
- Amplitudes de las impulsos:
- Tiempo entre impulsos.

Las dos primeras dependen del fabricante y de la carga del transformador, mientras que el retardo en el tiempo entre tramas depende del número de operaciones en el OLTC.

1.4. FASES DEL PROYECTO.

Las distintas fases se desarrollan en paralelo a los objetivos marcados.

Como se introdujo anteriormente, se hace uso de Matlab como herramienta de desarrollo, debido a su capacidad de cálculo y a las herramientas que posee, ideales para el análisis a desarrollar.

Es inevitable pasar un tiempo en busca de la base teórica que permita la implementación de la transformada de wavelet de la manera más sencilla y eficaz posible. Además dado que existen implementaciones en matlab para el análisis de la señal, por medio de la transformada de wavelet (wavemenu), se estudiará su funcionalidad para tomar las ventajas y eliminar los inconvenientes de la misma, de cara al objeto de etudio.

En la primera fase del proyecto, se tratará de la implementación de los métodos matriciales adecuados, que permita el análisis de la señal con los distintos tipos de wavelet madres. Dado que la señal de entrada posee una gran número de muestras (500000) será imposible de tratar mediante métodos matriciales, lo que obliga al estudio de la señal de entrada, para poder comprimir la información.

Una vez se tenga la información de la señal en un menor número de muestras y se halla aplicado el método de análisis desarrollado, se extraerán las características que hacen posible la clasificación de la avería.

Se elaborará posteriormente un método que permita una mejora en la selección de los coeficientes obtenidos mediante la transformada de wavelet, para de esta forma, mejorar la detección de los impulsos, y permitir una posterior recuperación de la señal.

Además del cálculo de la correlación lo cual permite tener un valor de medida de la fiabilidad del método, se desarrolla una segunda herramienta de análisis, que hará uso de las

funciones propias de Matlab, para comparar los resultados con el método matricial (además este método permitirá un futuro análisis frecuencia-tiempo de la señal de entrada).

Tras la correcta obtención de los coeficientes, se extraerán las características pertinentes de la señal para la comparación de ambos métodos.

Por último se realiza la interfaz, la cual permite la interacción del usuario con la herramienta de análisis desarrollada, así como la introducción de parámetros que caractericen el análisis y la presentación de los resultados.

Tras el desarrollo completo de la herramienta, se optimizará y se permitirán ciertas opciones, como pueden ser la impresión de los resultados en un documento, el salvado de los coeficientes, la presentación de la interfaz, etc…

El desarrollo de un manual de usuario, así como la documentación tanto del código como del proceso global, compondrá el broche final del proyecto.

1.5. ESTRUCTURA DEL DOCUMENTO.

El desarrollo del proyecto se llevará a cabo bajo la división de diferentes capítulos y un anexo final.

CAPITULO 1: Explicación general de los motivos y objetivos que llevan al desarrollo del trabajo, así como su enfoque dentro de un estudio previo y posterior, y las diferentes fases que lo componen.

CAPITULO 2: Describe el tratamiento de las distintas mediciones obtenidas del OLTC, para su posterior uso, con el objetivo de minimizar el tiempo de cálculo (coste computacional). De este modo se trata la señal de entrada para garantizar un adecuado pos procesado, tanto con la transformada de wavelet, como en una futura detección de impulsos.

En este apartado se detalla los diferentes métodos usados para optimizar la relación muestras e información, haciendo lo más eficiente posible las fases posteriores del trabajo.

CAPITULO 3: Análisis en detalle de la transformada de wavelet y análisis multiresolución, en este apartado se explicará la obtención de los coeficientes para los diferentes tipos de wavelets madre. Se detallará el análisis de la obtención de los coeficientes, haciendo hincapié en el desarrollo matricial, para un posterior tratamiento y recuperación de la señal.

CAPITULO 4: Análisis de los resultados, comparación, consecuencia del uso de un método u otro y conclusiones.

CAPITULO 5: Desarrollo de la interfaz y manual de usuario

CAPITULO 6: Conclusiones.

CAPITULO 7: Trabajos futuros.

CAPITULO 2. TRATAMIENTO DE LA SEÑAL DE VIBRACION

A lo largo de este apartado se establecerá un orden, donde se observa cada una de las fases del tratamiento de la señal. Pese a que posteriormente se verá más en detalle, que este análisis puede incorporar ciertas variaciones, en este caso se indica en líneas generales el tratamiento de partida de la misma, así como la descripción del método matricial que acompaña al resto del análisis.[1]

Los rasgos generales del análisis componen las etapas descritas y mostradas en la ilustración3:

- ✓ Cálculo de la señal en energía y análisis de la toma de intervalos óptimos para dicho computo.
- ✓ Primera detección de los impulsos de la señal.
- ✓ Eliminación de los márgenes de la señal que no aporten información.
- ✓ Análisis multirresolución (aplicación del método matricial de la transformada de wavelet)
- ✓ Selección de coeficientes y detección de los impulsos
- ✓ Extracción de los indicadores de diagnostico.

[1] Hacemos referencia a este hecho, dado que posteriormente se observará otro tipo de análisis (llevado a cabo mediante funciones implementadas por matlab

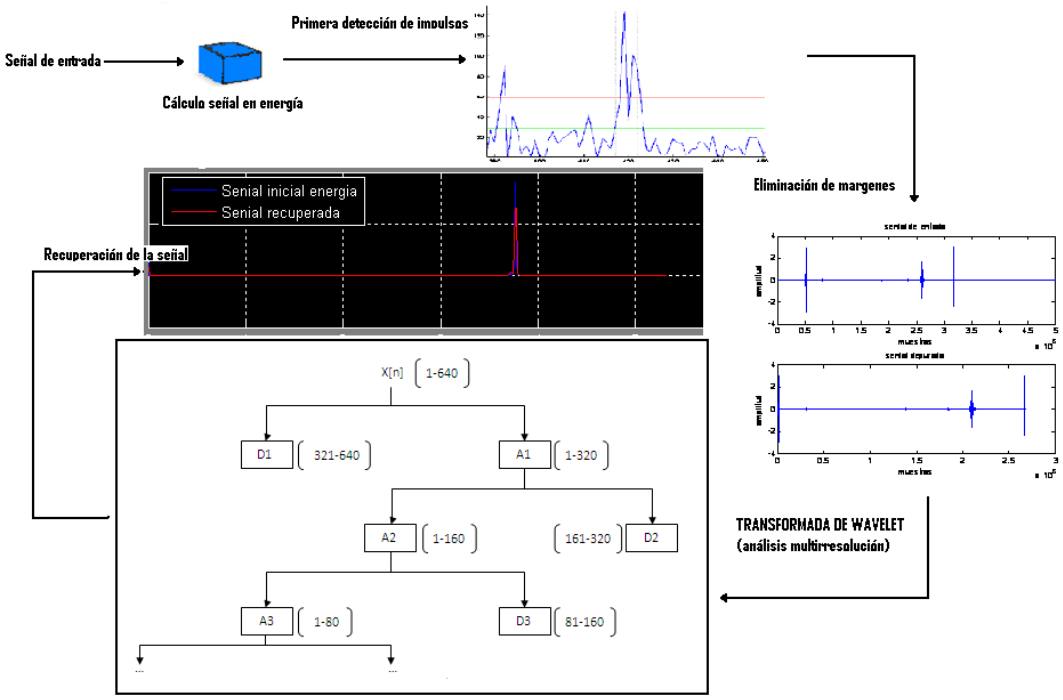


Ilustración 3. Tratamiento de la señal de vibración

2.1. INTRODUCCIÓN

La principal problemática que posee la señal a tratar es la gran cantidad de muestras que posee (500000) lo cual, hace imposible su tratamiento directo con la transformada de wavelet (implementación matricial), ya que requeriría tener matrices de cálculo de 500000 filas, lo cual requiere una capacidad de memoria muy alta. (No compatible con el tratamiento que se propone)

Es por esto que se debe desarrollar un mecanismo previo, que permita el tratamiento de las muestras, con el objetivo de alterar lo menos posible, la información que permite clasificar la señal para su diagnostico.

2.2. ELIMINACION DE LOS MARGENES DE SIN INFORMACIÓN.

Eliminar las muestras que carecen de información, es la tarea principal de este primer paso, previo al análisis con la transformada de wavelet.

Dado que la finalidad es, la detección de las vibraciones, tendremos en cuenta solamente las muestras que aparezcan entre el primer y el último pulso, es decir, todas las muestras previas al primer impulso y posteriores al último serán eliminadas, ya que no aportan información.

Como efecto de este paso, obtenemos una reducción que va de 267001 a 269001 muestras (dependiendo de la localización de los impulsos de la señal de

entrada) lo que equivale a un margen de reducción de las muestras de entre el 53.40% y el 53.80% lo cual indica la gran importancia de este pre-procesado, mostrado en la ilustración4 (se muestran dos señales como ejemplo).

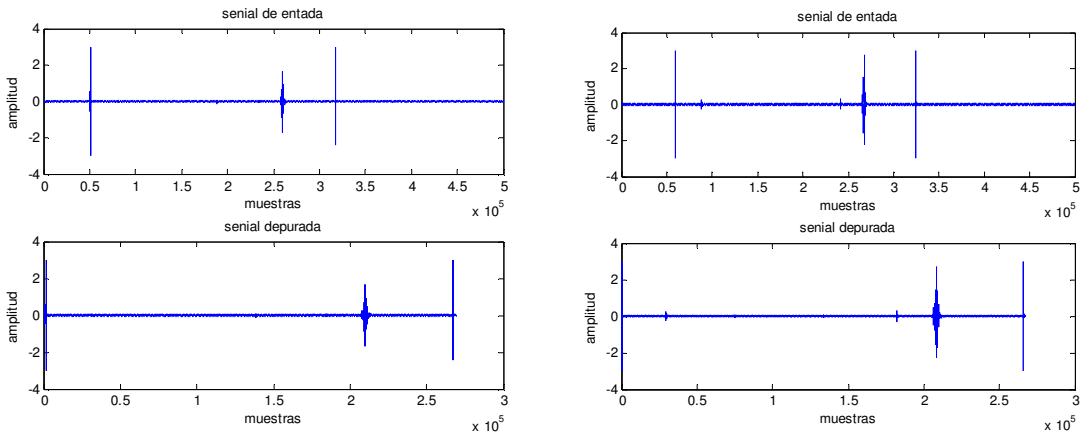


Ilustración 4. Eliminación márgenes de la señal de entrada

Para este paso previo, se ha realizado una primera detección de los impulsos de la señal (haciendo para ello un cálculo de la energía de la misma y de un método de selección de muestras a usar).

2.3. ANÁLISIS GENÉRICO: DETECCIÓN DE LOS IMPULSOS, USO LA SEÑAL EN ENERGÍA.

La detección de impulsos compone uno de los bloques principales en el desarrollo de la herramienta de análisis, para lo que es necesario un estudio exhaustivo de las características de las señales de entrada, así como de la elección del método más adecuado.

Se tienen dos parámetros clave, el primero, el umbral de detección, que se obtiene analizando cuál será el impulso más pequeño de las señales promedio (y apoyándonos en la ilustración 6), y los intervalos en que es dividida la señal en energía. A tal efecto se desarrolla un método, que logra fijar el número de intervalos en energía óptimos en los que se divide la señal de entrada, para la detección de los impulsos. Este número de intervalos, se ha seleccionado realizando el proceso completo con señales promedio de diferentes señales medidas, (el número de muestras de partida es 625).

Dado que la señal obtenida en un OLTC varia pese a realizar la misma maniobra dos veces, los impulsos estarán en posiciones distintas y no habrá dos realizaciones iguales. Llegados a este punto se plantean varios problemas:

- El estudio debe hacerse lo más genérico posible, debiendo de corresponderse de forma aproximada a las características de las distintas señales de entrada. Por lo que se realiza el entrenamiento con señales promedio de las medidas obtenidas. (De donde se obtiene un muestreo por defecto de 625 muestras, como se comentó anteriormente).

- Al realizar el promedio directamente de las señales de entrada, la localización de los impulsos estará en instantes de tiempo distintos, lo que dará lugar a diagnósticos erróneos, por lo que se debe de sincronizar las distintas señales para no detectar más impulsos de los que se tendría entrenando la máquina con las señales por separado. Además, se seleccionan señales que posean características similares, para evitar errores en las detecciones.
  Aun así, destacar que este entrenamiento se realiza tan sólo, para tener una estimación aproximada del número de muestras en energía a tomar, de la señal de entrada, necesarias para la detección de los impulsos.

  La solución adoptada, de tomar la energía de la señal para la detección de los impulsos y para la compresión de la información, teniendo por tanto un menor número de muestras, requiere un estudio detallado de los pros y contras, ya que en esta fase, se está eliminando información de la señal.

A) VENTAJAS:

Dado que a la hora de detectar los impulsos, la amplitud de la señal es, la característica principal, parece evidente su uso como herramienta preliminar a la transformada de wavelet.

Tras el proceso previo, donde las señales pueden contener distinto número de muestras, el objetivo es la obtención de una señal con unas características similares para cualquiera que sea la señal de entrada.

Desarrollando un método que permita la correcta detección de los 6 impulsos al inicio, variando los intervalos a tomar de la señal de entrada (para calcular la energía de la señal por intervalos), se asegura buena parte del éxito en el análisis.

Al tener una señal perfectamente caracterizada, podemos trabajar con el promedio de la señal en energía.

B) DESVENTAJAS, PROBLEMAS A RESOLVER

Se debe hacer un estudio sobre el número de intervalos que se requieren para que el muestreo conjunto con la detección, sea óptima. Se ha fijado una

compresión máxima, es decir, para todas las señales de entrada, se han tomado un número mínimo de muestras en energía, que permitan la detección.

Con este método, pese a mantener la información necesaria para la detección, se está filtrando la señal, obviando rangos de frecuencia de la misma. Con este "suavizado de la señal" pese a realizarse un estudio previo, el cual intenta evitar que se pierda información, se elimina un amplio margen de rangos de frecuencia en la señal.

Es por esto que se ha fijado una frecuencia de muestreo mínima, y un método de aumento de dicha frecuencia, para evitar eliminar la información imprescindible.

El desarrollo de este proceso previo, permitirá un margen de maniobra pequeño a la transformada de wavelet, dado que con el método matricial se parte de una señal depurada. Es por esto que se adjunta un método basado en funciones wavelet de matlab, que permitirá un análisis frecuencial, previo al cálculo de la señal en energía.

Se parte de la señal representada en la figura2.1. la cual se halla en primer lugar la energía, tomada por tramos, de cada una de las señale a la entrada, y posteriormente se toma una señal promedio para hacer una mejor caracterización en apartados posteriores.

La elección de los intervalos en que quedará dividida la señal de energía tiene una vital importancia, ya que de este parámetro junto con la elección del umbral dependerá el error en la detección.

Las fases de procesado de la señal en energía, para su posterior análisis, serán:

- Normalización de la señal, dada la gran variación que se puede dar en amplitud y tiempo entre impulsos.

$$S[n] = \frac{señal[n]}{\max(señal[n])}$$

Ecuación 1

- En condiciones normales debería haber una etapa previa de sincronización, pero dado que se tiene en cuenta la señal promedio, una vez detectado el primer pulso de forma individual para cada señal, la sincronización es inmediata.

  Esto es debido a que la señal a analizar comenzará a partir de la detección del primer impulso de cada una de las señales de entrada.

- Para evitar problemas en la detección de los impulsos se acude a la envolvente de la señal, ya que de modo contrario se podrían localizar impulsos de forma errónea.

  Para ello se usa la transformada de Hilbert, donde la parte real de la señal analítica es la señal original y la parte imaginaria es la transformada de Hilbert. La

magnitud de esta señal compleja forma la envolvente de la señal, que es siempre una señal positiva.

En la ilustración5 se representa el efecto de la transformada para una determinada señal de entrada. (Señal representada en el dominio del tiempo teniendo en cuenta una frecuencia de muestreo de la tarjeta de adquisición de 50Kmuestras/s).

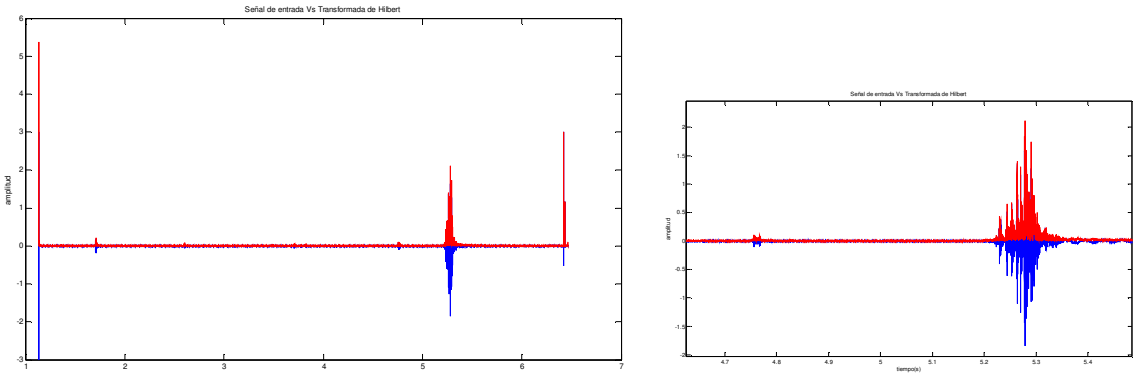


Ilustración 5. Cálculo de la señal en energía

Estrictamente la transformada de Hilbert no se hace para la primera detección de impulsos (cuando se depura la señal de entrada para detectar que muestras no contienen información) ya que el la vibraciones inicial y final no presentan problemas para ser detectadas. Luego se usa para una segunda etapa, donde el proceso se hace más complicado.

Tras el análisis de la selección de los intervalos (con un umbral seleccionado previamente mediante inspección y uso de Ref. [5]), se considera maximizar la cantidad de energía conservada de la señal, y minimizar el error cuadrático medio, mostrado en la ilustración6.

Con este análisis se pretende establecer un intervalo en el que situar el umbral de detección duro. La región lineal (donde el error MSE se mantiene a 0) queda caracterizada hasta un umbral de 1.3 aproximadamente, es entonces este intervalo sobre el que se implementará el método de detección de umbral óptimo.

Una vez se seleccionan diversos valores del umbral de detección de impulsos se estudian las señales promedio para cada uno de los impulsos que se deben detectar, en especial, aquel impulso con menor amplitud y se selecciona el umbral adecuado.

Los parámetros analizados son:

- Energía preservada de la señal (Ep): Mide la porción de la energía preservada tras la aplicación del umbral.
- Error cuadrático medio (MSE): Este indicador es proporcional a la distorsión introducida por el umbral, por lo tanto debe ser lo más bajo posible.
- Relación señal a ruido (SNR): Este indicador mide la proporción de ruido presente en la señal.

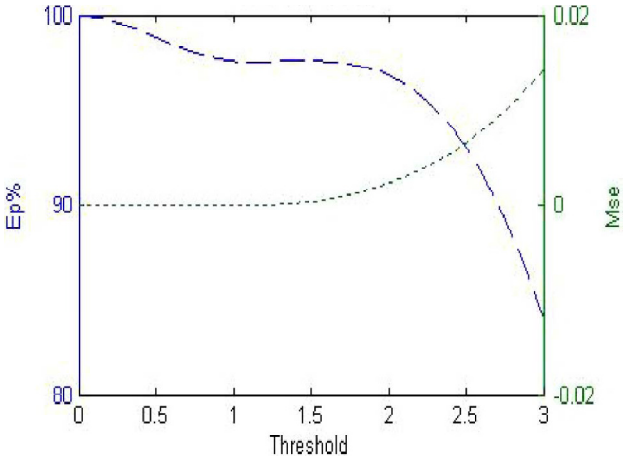


Ilustración 6. Energía almacenada de la señal vs error cuadrático medio (Ref [5])

$$Ep = \frac{\sum_{i=1}^{N} env_{th}[i]^2}{\sum_{i=1}^{N} env[i]^2} \cdot 100$$

Ecuación 2 (energía preservada de la señal)

$$MSE = \frac{1}{N}\sum_{i=1}^{N}\left[env[i] - envs[i]\right]^2$$

Ecuación 3 (error cuadrático medio)

$$SNR(db) = 10\log\frac{\sum_{i=1}^{N}[env[i]]^2}{\sum_{i=1}^{N}\left[env[i] - envs[i]\right]^2}$$

Ecuación 4 Relación señal a ruido

Tras aplicar el análisis anterior, se decidió fijar el umbral de forma que se pudieran detectar los impulsos para todos los casos. Es por esto que se desarrolla un método que permite detectar el umbral que nos proporciona mejores resultados, para posteriormente fijarlo en la etapa de detección.

- Una vez seleccionada la longitud de los tramos de la señal en energía se va recorriendo la señal y hallando la energía tramo a tramo.

En la ilustración7 se representa el efecto que tiene este proceso sobre la señal de entrada promedio.

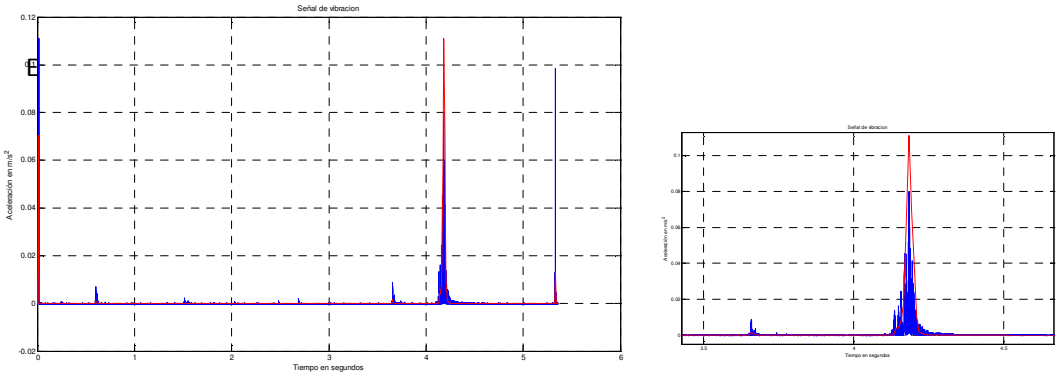


Ilustración 7. Procesado señal entrada promedio, y ampliación de los impulsos

- Llegado a este punto se tiene una señal de 625 muestras (y un umbral de detección de 0.003) con la que trabajar con la transformada de wavelet, de forma

que se pueda conseguir una reducción aún mayor de las muestras a utilizar para una posterior recuperación de las características principales necesarias de la señal de entrada. Este proceso, el cual compone el esqueleto principal del trabajo será desarrollado en el siguiente apartado.

2.3.1. Análisis particular: selección de intervalos a usar con la señal de energía.

Una vez se tiene el desarrollo genérico, para el cual se detectan los impulsos de la señal a tratar, tomando 625 muestras, se tiene que existen casos particulares donde este muestreo, no es suficiente para detectar la totalidad de los impulsos deseados.

Es por esto, que pese a hacer la totalidad del análisis, partiendo de la presunción, de que tenemos 625 muestras, se realiza previamente la detección del número de muestras idóneo para obtener los 6 impulsos.

Para ello se usa un método recursivo, donde se incrementa el número de intervalos a usar, hasta que se detectan los 6 impulsos, una vez aquí, se almacena el número de muestras que se van a tomar y se pasa a realizar el análisis.

Destacar por último que estas muestras corresponden a la totalidad de la señal, luego una vez se eliminan los márgenes previos y posteriores al primer y último impulso respectivamente, el número de muestras es aún menor.

CAPITULO 3. TRANSFORMADA DISCRETA DE WAVELET, ANÁLISIS DE MULTIRRESOLUCIÓN.

Como se mencionó anteriormente se usa el análisis multirresolución el cual analiza una señal para diferentes frecuencias con diferentes resoluciones.

El análisis MRA nos proporciona una pobre resolución en frecuencia y una buena resolución temporal, para altas frecuencias y buena resolución en frecuencia y baja en el tiempo para bajas frecuencias. Es por esto que en el caso que los planteamos sea de vital importancia su uso, ya que la señal a analizar tiene componentes de alta frecuencia de corta duración y componentes de baja frecuencia de larga duración.

Lo que subyace dentro de la transformada, es en realidad un filtrado de la señal en el tiempo mediantes filtros paso alto (señal de detalle) y paso bajo (señal de aproximación) que eliminan ciertas componentes de la señal, repitiéndose el proceso hasta el nivel seleccionado. De esta forma en nuestro caso, al tener una señal muestreada a 50KHz en su primera etapa se divide por un lado de 0 a 12.5KHz y por otro de 12.5 a 25KHz, tras esta primera etapa se suele tomar la señal a bajas frecuencias para pasar al siguiente nivel y continuar el proceso, aunque también se puede tomar la señal en su totalidad, de forma que tendremos la señal dividida de 0 a 6.25KHz de 6.25 a 12.5 de 12.5 a 1875KHz y de 18.75 a 25KHz. (En este análisis frecuencial no se ha tenido en cuenta el pre procesado, lo cual se hace para el punto 3.1.)

Las características principales de estos filtros son:

- Filtro mitad de banda, que dejan pasar solamente la mitad de la banda superior o inferior de la señal de entrada
- Ganancia DC igual a $\sqrt{2}$
- Los filtros de descomposición y reconstrucción son de cuadratura, lo que permite la recuperación de la señal.

Llegados a este cierto nivel de análisis, se puede representar la señal de forma que se tenga un grupo de señales pertenecientes a la misma señal pero correspondientes a distintas bandas de frecuencia. Esto hace pensar en la obtención de una señal tridimensional que contenga todas las posibles componentes de la señal y se visualice que bandas de frecuencias existen en un momento determinado.

He aquí la principal ventaja de la transformada de wavelet frente a la transformada de Fourier, ya que el análisis de esta última proporciona una resolución fija para todos los tiempos, mientras que la TW trabaja con una resolución variable.

En este caso de estudio, el desarrollo es realizado con la transformada de wavelet discreta, ya que trabajamos con señales de este tipo, lo cual proporciona una mayor facilidad de cálculo.

Matlab dispone de sus funciones propias para calcular la transformada de wavelet, sin embargo, como se comentó, las funciones son desarrolladas en su totalidad mediante operaciones matriciales, característica que hacía inviable el trabajo con la totalidad de las muestras de la señal, por lo que se optó por el procesado previo. Otro debate podría ser las ventajas que aporta el trabajar con funcionen implementadas por Matlab o por el usuario.

Durante el capitulo se trata el desarrollo que acompaña a cada tipo de transformada, así como sus principales características y los resultados que proporcionan de cara a un correcto análisis de las características de la señal en cuestión.

3.1. ANALISIS DE LA TRANSFORMADA DE WAVELET DISCRETA

3.1.1. Obtención de coeficientes.

(Ref. [2]) En primer lugar destacar la ventaja del uso de la transformada discreta, ya que la transformada entrega una información altamente redundante para la reconstrucción de la señal, lo que provoca un aumento significativo en el tiempo de cálculo.

De esta forma se tiene la suficiente información tanto para el análisis como para la reconstrucción de la señal con una significativa reducción del tiempo de procesamiento, e incluso con una mayor facilidad de implementación.

Dada la señal discreta S[n] su transformada viene dada por la ecuación 5:

$$C[j.k] = \sum_{Z} S[n] \cdot \Psi_{j,k}[n]$$

Ecuación 5

Donde $\Psi_{j,k}$ es una wavelet discreta definida por la expresión 6:

$$\Psi_{j,k} = 2^{\frac{1}{2}} \cdot \Psi[2^{-2j} n - k]$$

Ecuación 6

La transformada inversa se define de forma similar con la ecuación 7:

$$f[n] = \sum_{z} \sum_{z} C[j,k] \Psi_{j,k}[n]$$

Ecuación 7

El procedimiento descrito, puede ser representado a través de la descomposición de la ilustración3.1., donde las muestras aparecerán reseñadas en cada uno de los pasos.

Recordar que nuestra señal de energía por defecto tenía un total de 625 muestras, sin embargo, para trabajar con la transformada se necesita un múltiplo de $2^{niveles}$ de muestras, por lo que se añaden 15 muestras más a cero para hacer un total de 640. En caso de que la señal en energía tenga más de 640 muestras, no se realizará la transformada de wavelet por medio de cálculos matriciales.

En la ilustración8 se muestra un diagrama con el procesado de las muestras a los diferentes niveles. Donde D son las muestras en detalle, que se corresponden con las frecuencias más altas y A de aproximación se que corresponden a las frecuencias más bajas.

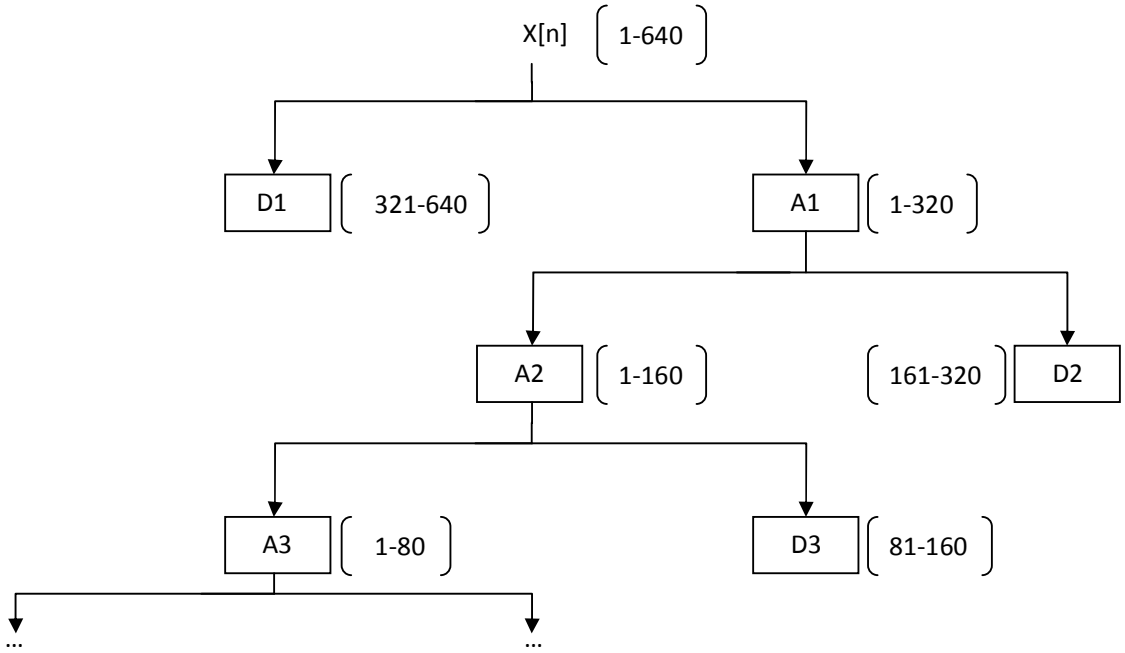


Ilustración 8.Diagrama de descomposición en muestras

Se ha tomado como ejemplo la señal de energía y se ha llegado a desarrollar hasta un nivel tres, pero como se verá, este parámetro se deja a libre elección del usuario.

Este análisis que en el ejemplo se ha dado en muestras, se puede realizar también atendiendo a la frecuencia de la señal. Con lo cual se tendría, que en cada una de las bifurcaciones existe un filtro paso alto y un paso bajo que permite el paso de la mitad superior e inferior respectivamente, esto es, suponiendo en nuestro caso, BW=25KHz (ya que como sabemos $f_{muestreo} = 50Khz$ y por Nyquist ese puede ser el máximo ancho de banda), tendremos una esta señal dividida en el primer nivel en 1-125 KHz y 125-2.5 KHz.

Este análisis es válido para antes del pre-procesado, ya que se tiene para un periodo de medida de 10sec un total de 500000 muestras. Sin embargo posteriormente se eliminan las muestras anteriores y posteriores al primer y último impulso respectivamente, quedando pues de 267001 a 269001 muestras (dependiendo de la señal analizada).

Aunque no es un muestreo real de la señal de partida, podemos considerarlo de este modo, de manera que se tiene una reducción en muestras de [267001,269001] a 625, con lo que tenemos la tasa mostrada en la ecuación 8.

10 seg → 500000 muestras.

X seg → [267001,269001]

Luego tenemos una señal de [5.34, 5.38] seg

$$f_{muestreo} = \frac{625}{[5.34,5.38]} \approx [116,117]Hz$$

Ecuación 8

Esto se puede resumir en una señal de 5.36 segundos aproximadamente y 625 muestras. Lo que supone una frecuencia máxima de los componentes de nuestra señal de 58.5Hz.

Este proceso continúa siempre que se tenga un número de muestras entero, que nos permita recuperar la señal. Para el ejemplo anterior se podrían dar hasta 7 niveles diferentes. Ya que $\frac{640}{2^7} = 5$.

Previo al análisis de cada uno de los tipos de wavelets madre a tratar se hará referencia al análisis y desarrollo seguidos para dar lugar a la

descomposición de coeficientes, basado en el tratamiento matricial de los datos. A continuación se destacan los rasgos generales comunes a todos ellos.

3.1.1.1. Desarrollo matricial genérico (Ref [1])

En cada una de los tipos ha de tenerse en cuenta que la descomposición de la señal en coeficientes se corresponde a la ecuación 9:

$$f \xrightarrow{H_m} \left(a^m \mid d^m \mid d^{m-1} \mid \ldots \mid d^1\right)$$

Ecuación 9

En la ilustración3.1 se tiene el proceso detallado de dicha descomposición, donde la señal es pasa por el filtro paso (en coeficientes de aproximación) y por el filtro paso alto (en coeficientes de detalle) en cada uno de sus niveles.

Dado que se tienen descomposiciones ortogonales, el proceso a seguir para hallarlos será el que sigue:

En primer lugar se obtienen los coeficientes de nivel 1 mediante la matriz de la ecuación 10:

$$\begin{bmatrix} a^1 \\ \hline b^1 \end{bmatrix} = \begin{bmatrix} v_1^1 \\ \vdots \\ v_{N/2}^1 \\ \hline w_1^1 \\ \vdots \\ v_{N/2}^1 \end{bmatrix} \times \begin{bmatrix} f_1 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ f_N \end{bmatrix}$$

La matriz asociada a dicha transformación hace que sus filas formen una base ortonormal, siendo a su vez H_1 ortogonal.

Esto proporciona una gran capacidad de cálculo ya que al realizar la transformada inversa se tiene:

$$\left(a^1 \mid d^1\right) \xrightarrow{H_1^{-1}} f$$

Ecuación 10

Dado el razonamiento anterior el cálculo de dicha transformada inversa resulta trivial, transponiéndose la matriz, por medio de la ecuación 11:

$$\begin{bmatrix} v_1^1 \\ \vdots \\ v_{N/2}^1 \\ \hline w_1^1 \\ \vdots \\ v_{N/2}^1 \end{bmatrix}^T \times \begin{bmatrix} a^1 \\ \hline b^1 \end{bmatrix} = \begin{bmatrix} f_1 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ f_N \end{bmatrix}$$

Ecuación 11

Tras las operaciones de nivel uno se pasa al cálculo de las matrices y coeficientes de niveles superiores.

Se presenta el cálculo de la transformada H_m obtenida de forma análoga, el cuál viene dado por la ecuación 12:

$$\begin{bmatrix} a^m \\ \hline d^m \\ \hline \vdots \\ \hline d^1 \end{bmatrix} = \begin{bmatrix} v_1^m \\ \vdots \\ \hline v_{N/2}^m \\ \hline w_1^m \\ \vdots \\ \hline w_{N/2}^m \\ \hline \vdots \\ \hline w_1^1 \\ \vdots \\ w_{N/2}^1 \end{bmatrix} x \begin{bmatrix} f_1 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ f_N \end{bmatrix}$$

Ecuación 12

Donde la misma forma en que se hizo con el primer nivel se puede obtener la transformada inversa.

Una vez detalladas las generalidades del método, se verá como cada una de las wavelet madre posee sus propias matrices que hace que cada uno de ellas tenga peculiaridades distintas.

Por ello el esqueleto del programa con el que se llevará a cabo el cálculo de los coeficientes será común para todos ellos, exceptuando por ejemplo el valor que tendrán los elementos de la matriz transformación, o la forma en que se irán construyendo las matrices de niveles superiores.

3.1.2. Extracción de los parámetros de diagnostico de la señal analizada.

Una vez descrito el procedimiento matricial para recuperar la señal a partir de los coeficientes de la transformada discreta de wavelet inversa, o en el método descrito, matriz generatriz inversa, se propone una mejora, de modo que se minimice el número de coeficientes a usar.

De esta manera, una vez se tengan los coeficientes de cada nivel, se pasa a intentar reconstruir la señal de partida, siguiendo el método tradicional de recomposición, y un segundo algoritmo:

1. En primer lugar, sabiendo que la señal se descompone en coeficientes de aproximación (frecuencias inferiores de la señal, donde existe una mayor cantidad de información) y coeficientes en detalle (frecuencias superiores donde se localizaran ciertas variaciones de la señal, o ruido), se trata de recuperar la señal con los coeficientes de aproximación.

2. En caso de no haber tenido éxito, se pasa al uso de los coeficientes en detalle del nivel actual, ya que puede ser que se halla obviado ciertas variaciones en frecuencia, que den lugar a un impulso.

3. Si de nuevo no se detectan los impulsos esperados, esto es debido a que ciertos impulsos pertenecen a frecuencias comprendidas en rangos superiores, luego debemos hacer uso de coeficientes en detalle de niveles superiores.

4. De esta forma vamos recorriendo el algoritmo hasta que se obtienen la totalidad de los impulsos que componen la señal analizada.

5. La ultima mejora se trata, de una vez se tiene la señal recuperada, tratar de obviar frecuencias intermedias que provienen de coeficientes en detalle de niveles intermedios, es decir, tras el algoritmo de búsqueda, desde el último, hasta el primer nivel analizado, se pasa a realizar el algoritmo en sentido inverso (iniciando desde el primer nivel desde donde se ha sido capaz de recuperar la señal) hasta el último. (Mejora no implementada)

Por último destacar, que el algoritmo de parada se establece, cuando se ha sido capaz de recuperar la totalidad de los impulsos de la señal de partida, extrayendo sus características.

- Instante de tiempo donde se localizan los impulsos.
- Tiempos entre impulsos.
- Amplitud (energía) de los impulsos.

3.2. ANALISIS DE LOS CASOS TIPICOS DE TRANSFORMADAS DE WAVELET

En el desarrollo del proyecto se analiza la señal de energía con distintas wavelet madre con el objetivo de seleccionar los coeficientes de aquella señal recuperada que mas se asemeje a la inicial.

Como paso previo al computo común de las matrices de cada uno de los tipos, se asignan unos valores iniciales dependiendo del tipo en cuestión lo que compone el aspecto diferenciador del método.

Sirva como ejemplo detallado la transformada de Haar, para posteriormente pasar a detallar los demás tipos de una forma más esquematizada.

3.2.1. Transformada de Haar

3.2.1.1. Descripción de la transformada

También conocida en algunos textos como Dauchebies con un solo nivel de desvanecimiento.

Es muy sencilla pero no por ello deja de ser una de las más usadas para el análisis de señales usando transformadas discretas y continuas.

Tiene como principales características que es ortogonal y biortogonal, se puede comprimir. Es simétrica, no regular, y admite soporte compacto. Es usada para la detección de discontinuidades y puntos de ruptura.

3.2.1.2. Obtención de coeficientes: Desarrollo matricial

En primer lugar, destacar que la selección de cada uno de los parámetros descritos, obedece a un objetivo, que no es otro que sub muestrear la señal, de modo que tengamos la mitad de muestras como coeficientes de aproximación, y la mitad de muestras como coeficientes en detalle (sumando entre ambas el mismo número de muestras que la señal a analizar).

Una vez descritas las directrices que permiten el cálculo de los coeficientes, y preparada la señal de entrada para el cálculo de los mismos, se pasa a la selección de los parámetros necesarios para hallar el tipo de transformada seleccionado:

- Selección de los parámetros a usar

En cada uno de los tipos se tienen como parámetros principales: los elementos iniciales para el cálculo de los coeficientes de nivel uno de la matriz de transformación y el desplazamiento que se lleva a cabo (descritos a continuación):

En este caso los parámetros a usar son:

$$\alpha = \left[\frac{1}{\sqrt{2}} \quad \frac{1}{\sqrt{2}} \right] \qquad \beta = \left[\frac{1}{\sqrt{2}} \quad \frac{-1}{\sqrt{2}} \right]$$

$$Desplazamiento = 2$$

- Construcción de la matriz transformación de nivel uno

Los parámetros que se obtienen para este primer nivel vienen dados por las ecuaciones 12 y 13:

$$a_1 = \frac{f_1 + f_2}{\sqrt{2}}, \quad a_2 = \frac{f_3 + f_4}{\sqrt{2}} \ \ldots \ a_{N/2} = \frac{f_{N-1} + f_N}{\sqrt{2}}$$

Ecuación 12

$$. \beta_1 = \frac{f_1 - f_2}{\sqrt{2}}, \quad \beta_2 = \frac{f_3 - f_4}{\sqrt{2}} \ldots \quad \beta_{N/2} = \frac{f_{N-1} - f_N}{\sqrt{2}}$$

Ecuación 13

Es por esto que las funciones de escala y las wavelet de Haar de nivel uno estarán formadas por los parámetros que se fijaron anteriormente, colocados como indica la ecuación 14:

$$\begin{aligned} v_1^1 &= [\alpha, 0, \ldots, 0] & w_1^1 &= [\beta, 0, \ldots, 0] \\ v_2^1 &= [0, 0, \alpha, 0, \ldots, 0] & w_2^1 &= [0, 0, \beta, 0, \ldots, 0] \\ &\vdots & &\vdots \\ v_{N/2}^1 &= [0, 0, \ldots, 0, \alpha] & w_{N/2}^1 &= [0, 0, \ldots, 0, \beta] \end{aligned}$$

Ecuación 14

Una vez en este punto se observa el por qué del parámetro $Desplazamiento = 2$ ya que en cada una de las filas de la matriz se ha desplazado α dos posiciones.

- Calculo de coeficientes y reconstrucción de la señal de nivel uno

Una vez hallada la matriz el cálculo de coeficientes se opera según la ecuación 15:

$$a_m = f \cdot v_m^1 \quad m = 1, \ldots, N/2. \qquad d_m = f \cdot w_m^1 \quad m = 1, \ldots, N/2.$$

Ecuación 15

Luego por el momento se tiene:

$$a^1 = (a_1, a_2, \ldots, a_{N/2}) \qquad d^1 = (d_1, d_2, \ldots, d_{N/2})$$

Y por tanto he aquí la primera reconstrucción de la señal como se describió en el desarrollo matricial del apartado 3.1.1.1. (según la ecuación 16):

$$f_{reconstruida} = \begin{bmatrix} v_1^1 \\ \vdots \\ v_{N/2}^1 \\ w_1^1 \\ \vdots \\ v_{N/2}^1 \end{bmatrix}^T \times \begin{bmatrix} a^1 \\ b^1 \end{bmatrix}$$

Ecuación 16

Esta es la señal con la cual, se irá trabajando para comprobar, usando los coeficientes de dicho nivel, si se puede recuperar la señal, detectando los impulsos de forma correcta.

- Calculo de la matriz del siguiente nivel

Se ha decidido operar sobre los parámetros α y β (como se muestra en la ecuación 17 y 18) cada vez que se incrementa el nivel de análisis, sirva como ejemplo el nivel dos:

$$\beta_{nueva} = \frac{1}{\sqrt{2^{nivel-1}}} \cdot [\alpha_{anterior} \quad -\alpha_{anterior}]$$

Ecuación 17

$$\alpha_{nueva} = \frac{1}{\sqrt{2^{nivel-1}}} \cdot [\alpha_{anterior} \quad \alpha_{anterior}]$$

Ecuación 18

A continuación se opera de forma que la matriz del siguiente nivel quede compuesta por la mitad de filas que el nivel anterior y mismo número de columnas, y con un desplazamiento de doble valor al anterior.

Poniendo el ejemplo del segundo nivel, la matriz quedará según las ecuaciones 19:

$$w_1^2 = \left(\frac{1}{2},\frac{1}{2},\frac{-1}{2},\frac{-1}{2},0,\ldots,0\right) \qquad v_1^2 = \left(\frac{1}{2},\frac{1}{2},\frac{1}{2},\frac{1}{2},0,\ldots,0\right)$$

$$w_2^2 = \left(0,0,0,0,\frac{1}{2},\frac{1}{2},\frac{-1}{2},\frac{-1}{2},0,\ldots,0\right) \qquad v_2^2 = \left(0,0,0,0,\frac{1}{2},\frac{1}{2},\frac{1}{2},\frac{1}{2},0,\ldots,0\right)$$

$$\vdots \qquad\qquad \vdots$$

$$w_{N/4}^2 = \left(0,\ldots 0,\frac{1}{2},\frac{1}{2},\frac{-1}{2},\frac{-1}{2}\right) \qquad v_{N/4}^2 = \left(0,\ldots 0,\frac{1}{2},\frac{1}{2},\frac{1}{2},\frac{1}{2}\right)$$

Ecuación 19

- Obtención de los nuevos coeficientes

Este apartado es idéntico a cuando obtuvimos los coeficientes del primer nivel, multiplicando la matriz generatriz por la señal.

Tras esto se vuelve a construir la matriz y se opera matricialmente hasta el nivel que se considere oportuno.

Después del desarrollo se tienen los coeficientes necesarios para la reconstrucción de la señal y su posterior evaluación.

Se deberá usar de una forma apropiada el desarrollo logrando minimizar los coeficientes para la recuperación de la señal.

Finalmente, la señal recuperada debe ser tratada para la detección de los impulsos.

3.2.1.3. Reconstrucción de la señal a partir de los coeficientes (primera aproximación)

Tras el análisis se comprueba, reconstruyendo la señal a partir de los coeficientes, que verdaderamente se tiene una buena aproximación de la señal, y lo que es más importante, una correcta detención de los impulsos.

Para ello se usan los coeficientes de menor a mayor nivel, hasta llegar a una expresión matricial dada por la ecuación 20:

$$\begin{bmatrix} f_1 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ f_N \end{bmatrix} = \begin{bmatrix} v_1^m \\ \vdots \\ v_{N/2}^m \\ \hline w_1^m \\ \vdots \\ w_{N/2}^m \\ \hline \vdots \\ \hline w_1^1 \\ \vdots \\ w_{N/2}^1 \end{bmatrix}^T x \begin{bmatrix} a^m \\ \hline d^m \\ \hline \vdots \\ \hline d^1 \end{bmatrix}$$

Ecuación 20

En la ilustración9 se muestra el análisis de la transformada descrita mediante la interfaz. Como vemos en la imagen queda reflejado:

- ✓ Nombre y tamaño de la señal analizada
- ✓ Tipo de wavelet
- ✓ Nivel
- ✓ Señal optimizada
- ✓ Número de impulsos
- ✓ Correlación

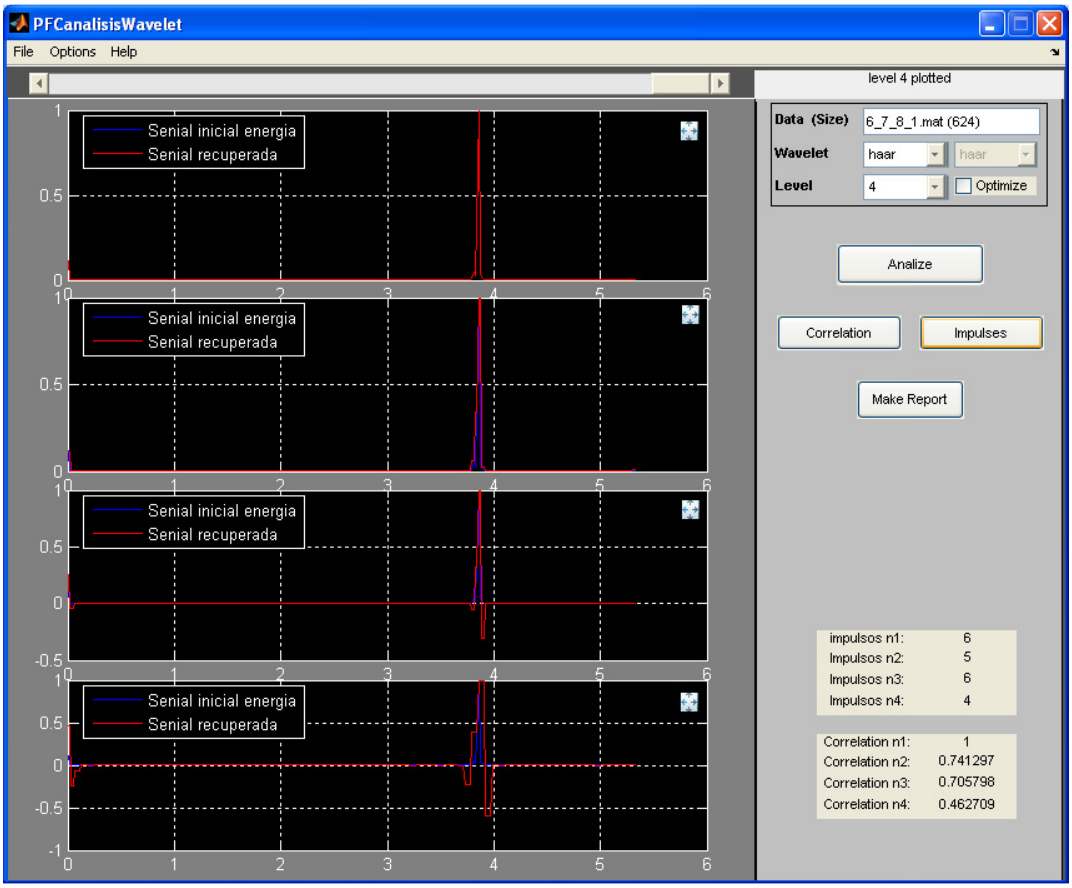


Ilustración 9. Análisis de la transformada de Haar

En la ilustración10 se muestra la señal de nivel dos amplificada, donde se puede observar los impulsos:

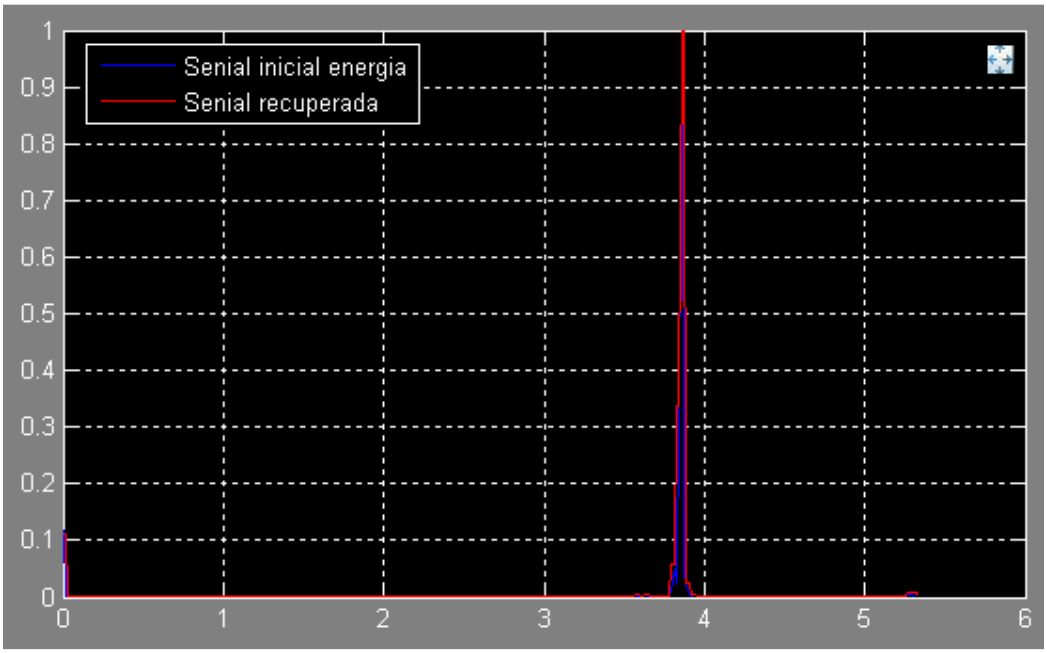


Ilustración 10. Ampliación del nivel 1

Quedan guardadas las características de los impulsos, comparadas al final de este apartado.

Se muestra en la ilustración11 el análisis de la misma señal mediante el método de optimizado[2]

[2] Consultar especificaciones de la interfaz

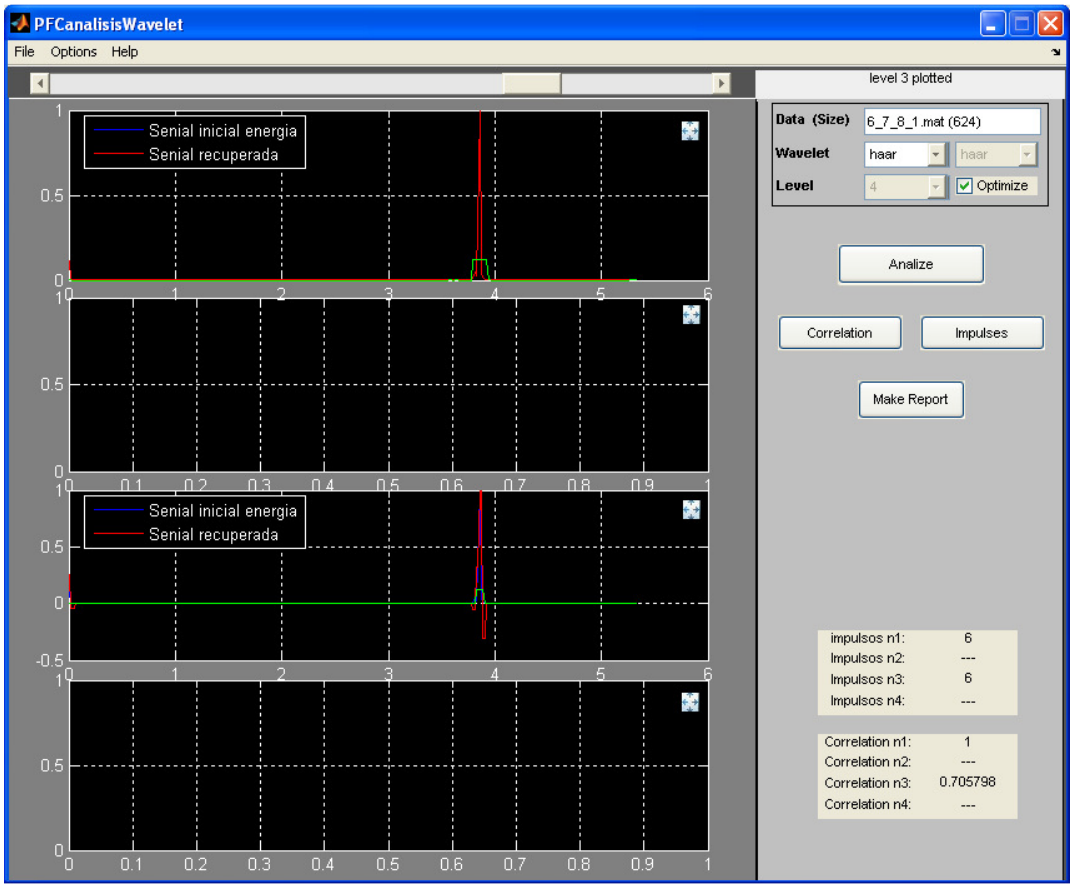


Ilustración 11. Optimización del análisis

Como se observa en la ilustración11, tan sólo se muestra la señal en caso de que se recuperen los 6 impulsos, en la ilustración12 vemos ampliada la señal de nivel 2:

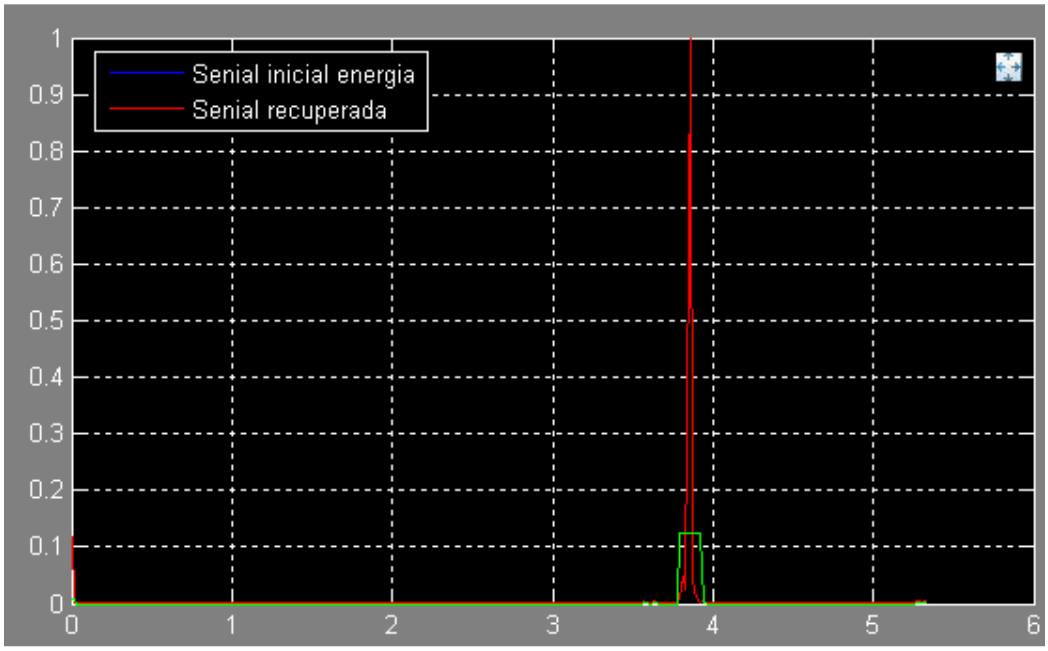


Ilustración 12. Ampliación de la señal de nivel 2

Se pasa a analizar otra señal, en este caso la 6_7_8_2.mat, mostrada en la ilustración13 y optimización mostrada en la ilustración14 :

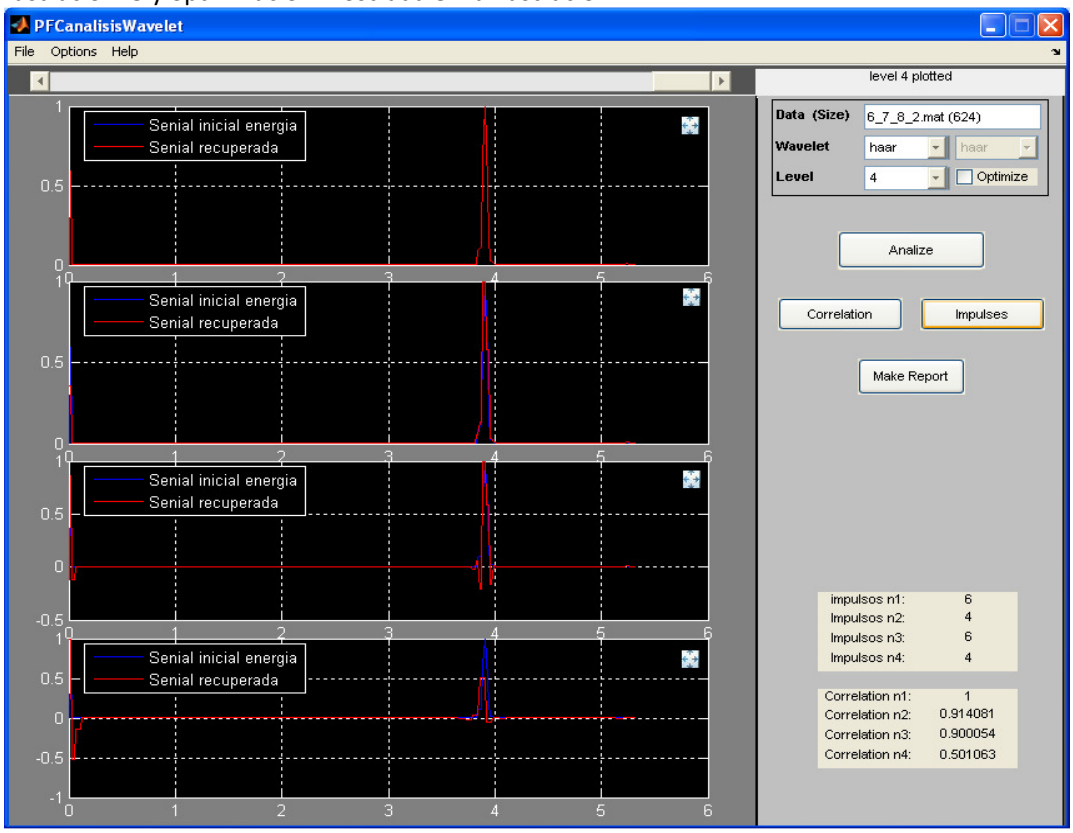


Ilustración 13. Análisis de la transformada de Haar

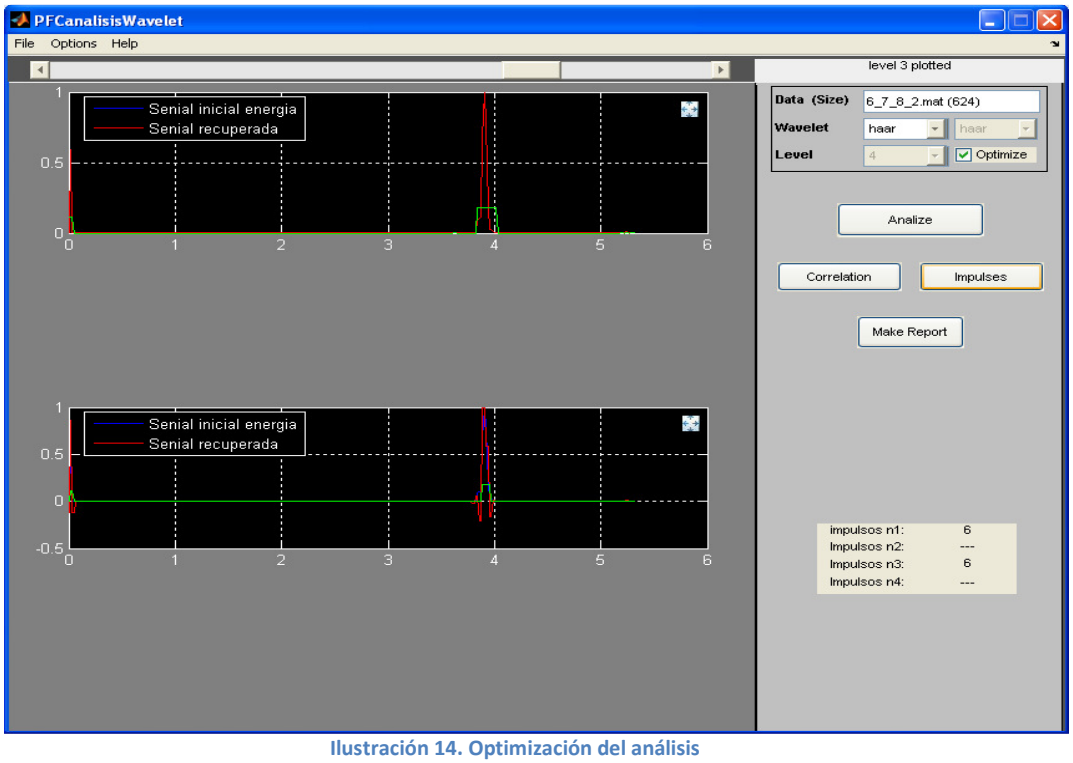


Ilustración 14. Optimización del análisis

3.2.2. Transformada de Daubechies dB2

3.2.2.1. Descripción de la transformada

La familia de las daubechies presenta unas características comunes y un número de orden que indicará la cantidad de momentos de desvanecimiento.

Admiten soporte compacto, son ortogonales y biortogonales, la mayoría no son simétricas, la regularidad va amentando con el orden y se puede usar para calcular tanto transformadas continuas como discretas.

3.2.2.2. Obtención de coeficientes: Desarrollo matricial

Siguiendo el método general, se toman los parámetros de entrada en concordancia al análisis actual.

- Selección de los parámetros a usar

Necesarios para la matriz de cálculo de coeficientes de nivel uno y el desplazamiento para niveles sucesivos.

En este caso los parámetros a usar son:

$$\alpha = \left[\frac{1+\sqrt{3}}{4\sqrt{2}} \quad \frac{3+\sqrt{3}}{4\sqrt{2}} \quad \frac{3-\sqrt{3}}{4\sqrt{2}} \quad \frac{1-\sqrt{3}}{4\sqrt{2}} \right]$$

$$\beta = \left[\frac{1-\sqrt{3}}{4\sqrt{2}} \quad \frac{-3+\sqrt{3}}{4\sqrt{2}} \quad \frac{3+\sqrt{3}}{4\sqrt{2}} \quad \frac{-1-\sqrt{3}}{4\sqrt{2}} \right]$$

$$Desplazamiento = 2$$

- Construcción de la matriz transformación de nivel uno

En este caso el algoritmo a usar es el mismo que en la construcción de la matriz de Haar, con la salvedad de que al ir desplazando la matriz de coeficientes de dos en dos, α es de tamaño doble al anterior mostrándose dicho análisis en la ecuación 21:

$$v_1^1 = (\alpha_1, \alpha_{2,} \alpha_{3,} \alpha_4, 0, 0, \ldots 0)$$
$$v_2^1 = (0, 0, \alpha_1, \alpha_{2,} \alpha_{3,} \alpha_4, 0, \ldots 0)$$
$$\vdots$$
$$v_{N/2-1}^1 = (0, \ldots, 0, \alpha_1, \alpha_{2,} \alpha_{3,} \alpha_4)$$

$$v_{N/2}^1 = (\alpha_3, \alpha_4, 0, ..., 0, \alpha_1, \alpha_2)$$

Mismo desarrollo para w, en la ecuación 21 (calculo matriz en diferencias)

$$\begin{aligned} w_1^1 &= (\beta_1, \beta_2, \beta_3, \beta_4, 0, 0, ...0) \\ w_2^1 &= (0, 0, \beta_1, \beta_2, \beta_3, \beta_4, 0, ...0) \\ &\vdots \\ w_{N/2-1}^1 &= (0, ..., 0, \beta_1, \beta_2, \beta_3, \beta_4) \\ w_{N/2}^1 &= (\beta_3, \beta_4, 0, ..., 0, \beta_1, \beta_2) \end{aligned}$$

Ecuación 21

- Calculo de coeficientes y reconstrucción de la señal de nivel uno

Una vez hallada la matriz el cálculo de coeficientes se opera de forma según la ecuación 22:

$$a_m = f \cdot v_m^1 \quad m = 1, ..., N/2. \qquad d_m = f \cdot w_m^1 \quad m = 1, ..., N/2.$$

Ecuación 22

Luego por el momento se tiene:

$$a^1 = (a_1, a_2, ..., a_{N/2}) \qquad d^1 = (d_1, d_2, ..., d_{N/2})$$

Y por tanto se llega a la primera reconstrucción de la señal como se describió en el desarrollo matricial del apartado 3.1.1.1. según la ecuación 23

$$f_{reconstruida} = \begin{bmatrix} v_1^1 \\ \vdots \\ v_{N/2}^1 \\ \hline w_1^1 \\ \vdots \\ v_{N/2}^1 \end{bmatrix}^T \times \begin{bmatrix} a^1 \\ \hline b^1 \end{bmatrix}$$

Ecuación 23

Esta es la señal con la cual, se irá trabajando para comprobar usando los coeficientes de dicho nivel, si se puede recuperar la señal, detectando los impulsos de forma correcta.

- Calculo de la matriz del siguiente nivel

Tanto para este caso como para los siguientes actuaremos de forma recurrente para el cálculo de las matrices de niveles mayores que uno.

De esta forma para hallar las siguientes matrices se usa la matriz del nivel anterior, dando luegar a los vecores formados por la ecuación 24:

$$v_1^m = \alpha_1 v_1^{m-1} + \alpha_2 v_2^{m-1} + \alpha_3 v_3^{m-1} + \alpha_4 v_4^{m-1}$$

$$v_2^m = \alpha_1 v_3^{m-1} + \alpha_2 v_4^{m-1} + \alpha_3 v_5^{m-1} + \alpha_4 v_6^{m-1}$$

$$\vdots$$

$$v_{N/2^m}^m = \alpha_1 v_{N/2^{m-1}-1}^{m-1} + \alpha_2 v_{N/2^{m-1}}^{m-1} + \alpha_3 v_1^{m-1} + \alpha_4 v_2^{m-1}$$

$$w_1^m = \beta_1 w_1^{m-1} + \beta_2 w_2^{m-1} + \beta_3 w_3^{m-1} + \beta_4 w_4^{m-1}$$

$$w_2^m = \beta_1 w_3^{m-1} + \beta_2 w_4^{m-1} + \beta_3 w_5^{m-1} + \beta_4 w_6^{m-1}$$

$$\vdots$$

$$v_{N/2^m}^m = \beta_1 w_{N/2^{m-1}-1}^{m-1} + \beta_2 w_{N/2^{m-1}}^{m-1} + \beta_3 w_1^{m-1} + \beta_4 w_2^{m-1}$$

Ecuación 24

- Obtención de los nuevos coeficientes

El cálculo de los coeficientes tanto para este, como para los niveles sucesivos, es trivial e igual a los hallados en el nivel uno.

Es evidente que una vez se tiene la matriz generatriz del nivel actual, tan solo es necesario multiplicar para la obtención de los coeficientes.

La condición de parada, tanto para este, como para el resto de los casos es, la selección de nivel a hallar seleccionada por el usuario, o en caso de haber usado la herramienta “optimizar” la detección de los seis impulsos y extracción de las características para la determinación del grupo al que pertenece la señal.

3.2.2.3. Reconstrucción de la señal a partir de los coeficientes (primera aproximación)

Tras el análisis se tiene, reconstruyendo la señal a partir de los coeficientes, una buena aproximación de la señal, y lo que es más importante, una (a priori) correcta detención de los impulsos.

Para ello iremos usando los coeficientes de menor a mayor nivel, hasta llegar a una expresión matricial dada por la ecuación 25:

$$\begin{bmatrix} f_1 \\ \vdots \\ f_N \end{bmatrix} = \begin{bmatrix} v_1^m \\ \vdots \\ v_{N/2}^m \\ w_1^m \\ \vdots \\ w_{N/2}^m \\ \vdots \\ w_1^1 \\ \vdots \\ w_{N/2}^1 \end{bmatrix}^T x \begin{bmatrix} a^m \\ d^m \\ \vdots \\ d^1 \end{bmatrix}$$

Ecuación 25

Las señales obtenidas tras el desarrollo descrito anteriormente se muestran en la ilustración15:

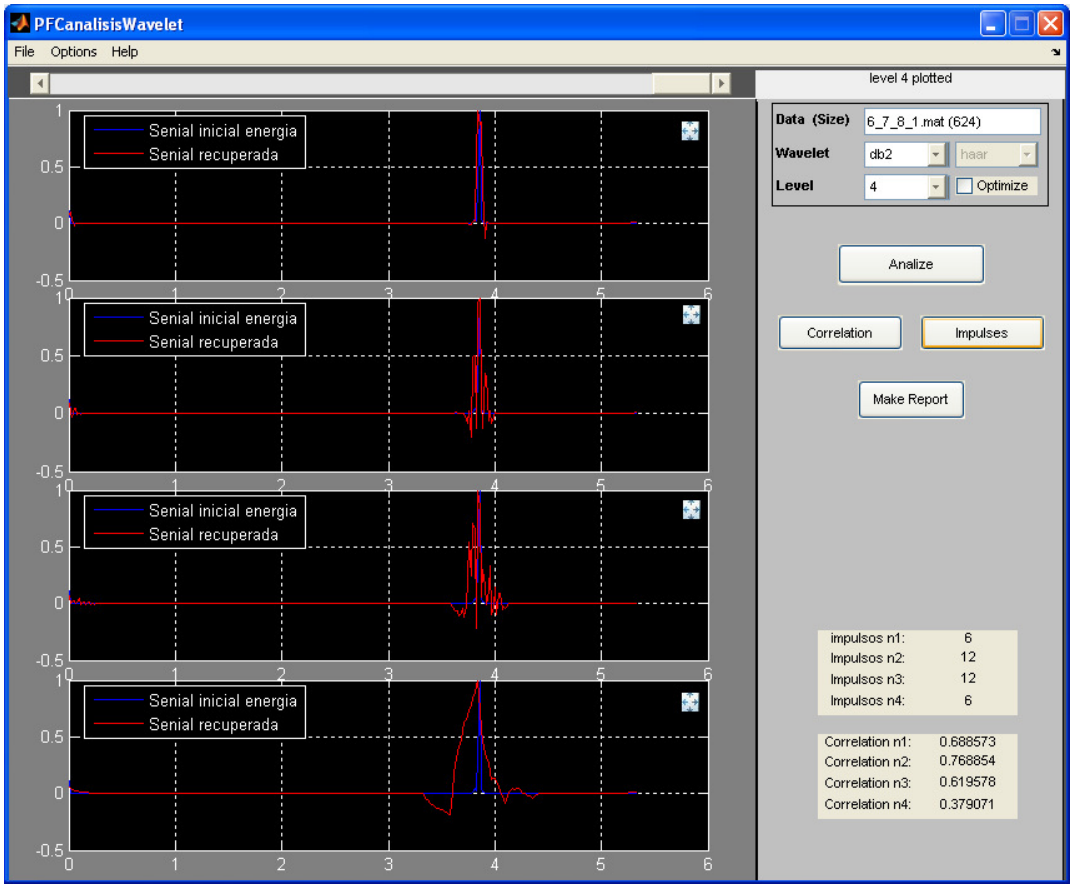


Ilustración 15. Análisis con la transformada de daubechies dB2

Como se puede observar reconstrucción de la señal con este tipo de transformada nos aporta una peor correlación que con la transformada de Haar.

La baja correlación alcanzada en el nivel 4, levanta sospechas acerca de la correcta detección de los impulsos (en la gráfica se puede apreciar que no va a ser correcta)

Usando otra señal de entrada se tiene los resultados mostrados en la ilustración16:

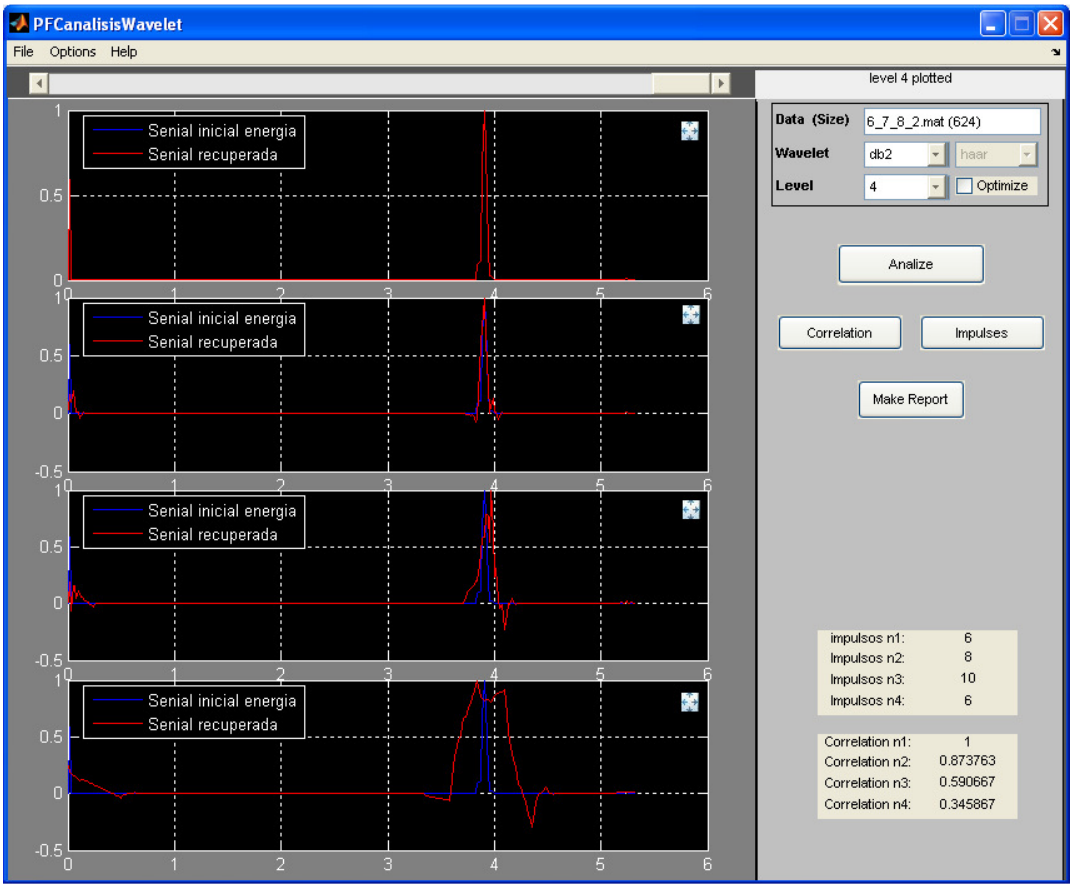


Ilustración 16. Análisis con la transformada de daubechies dB2

En este caso se obtienen 6 impulsos para el primer y cuarto nivel.

Dada la baja correlación de este ultimo, se debe desconfiar de la validez de la detección de los impulsos con el cuarto nivel.

3.2.3. Transformada de daubechies dB3

3.2.3.1. Descripción de la transformada

Al seguir en la familia de las Daubechies y tras haberse detallado anteriormente las características de dicha transformada se pasa directamente a la obtención de coeficientes.

3.2.3.2. Obtención de coeficientes: Desarrollo matricial

Tal como hicimos en el apartado anterior, deberemos en primer lugar seleccionar los parámetros adecuados.

- Selección de los parámetros a usar

Necesarios para la matriz de cálculo de coeficientes de nivel uno y el desplazamiento para niveles sucesivos.

En este caso los parámetros a usar son:

$$\alpha_1 = 0.332670552950083$$
$$\alpha_2 = 0.806891509311092$$
$$\alpha_3 = 0.459877502118491$$
$$\alpha_4 = -0.135011020010255$$
$$\alpha_5 = 0.0854412738820267$$
$$\alpha_6 = 0.0352262918857095$$

$$\alpha = [\alpha_1 \quad \alpha_2 \quad \alpha_3 \quad \alpha_4 \quad \alpha_5 \quad \alpha_6]$$
$$\beta = [\alpha_6 \quad -\alpha_5 \quad \alpha_4 \quad -\alpha_3 \quad \alpha_2 \quad -\alpha_1]$$

$$Desplazamiento = 2$$

- Construcción de la matriz transformación de nivel uno

Existen pocas diferencias con respecto a la formación del apartado anterior, tan solo cabe diferenciar los dos últimos casos en vez de tan sólo el último, de forma que la matriz de primer nivel queda según la ecuación 26:

$$v_1^1 = (\alpha_1, \alpha_{2,} \alpha_{3,} \alpha_4, \alpha_5, \alpha_{6,} 0, \ldots 0)$$
$$v_2^1 = (0,0, \alpha_1, \alpha_{2,} \alpha_{3,} \alpha_4, \alpha_5, \alpha_{6,} 0, \ldots, 0)$$
$$\vdots$$
$$v_{N/2-1}^1 = (\alpha_5, \alpha_{6,}, 0, \ldots, 0, \alpha_1, \alpha_2, \alpha_{3,} \alpha_4)$$
$$v_{N/2}^1 = (\alpha_{3,} \alpha_4, \alpha_5 \alpha_6, 0,0, \ldots 0, \alpha_1, \alpha_2)$$

Ecuación 20

Y se sigue el mismo desarrollo para w (calculo matriz en diferencias)

$$w_1^1 = (\beta_1, \beta_{2,} \beta_{3,} \beta_4, \beta_5, \beta_{6,} 0,...0)$$
$$w_2^1 = (0,0, \beta_1, \beta_{2,} \beta_{3,} \beta_4, \beta_5, \beta_{6,} 0,...,0)$$
$$\vdots$$
$$w_{N/2-1}^1 = (\beta_5, \beta_{6,}, 0,...,0, \beta_1, \beta_2, \beta_{3,} \beta_4)$$
$$w_{N/2}^1 = (\beta_{3,} \beta_4, \beta_5 \beta_6, 0,0,...0, \beta_1, \beta_2)$$

Ecuación 27

- Calculo de coeficientes y reconstrucción de la señal de nivel uno

Una vez hallada la matriz el cálculo de coeficientes se opera según la ecuación 28:

$$a_m = f \cdot v_m^1 \quad m = 1,..., N/2. \qquad d_m = f \cdot w_m^1 \quad m = 1,..., N/2.$$

Ecuación 28

Luego por el momento se tiene:

$$a^1 = (a_1, a_2,..., a_{N/2}) \qquad d^1 = (d_1, d_2,..., d_{N/2})$$

Y por tanto ya podemos hacer la primera reconstrucción de la señal como se describió en el desarrollo matricial del apartado 3.1.1.1. que viene dado por la ecuación 29

$$f_{reconstruida} = \begin{bmatrix} v_1^1 \\ \vdots \\ v_{N/2}^1 \\ w_1^1 \\ \vdots \\ v_{N/2}^1 \end{bmatrix}^T \times \begin{bmatrix} a^1 \\ b^1 \end{bmatrix}$$

Ecuación 29

Esta es la señal con la cual, se irá trabajando para comprobar usando los coeficientes de dicho nivel, se puede recuperar la señal, detectando los impulsos de forma correcta.

- Calculo de la matriz del siguiente nivel

Tanto para este caso como para los siguientes actuaremos de forma recurrente para el cálculo de las matrices de niveles mayores que uno.

De esta para hallar las siguientes matrices nos apoyaremos operando con la de nivel anterior, obteniendo los vectores mediantes la ecuación 30 y 31:

$$v_1^m = \alpha_1 v_1^{m-1} + \alpha_2 v_2^{m-1} + \alpha_3 v_3^{m-1} + \alpha_4 v_4^{m-1} + \alpha_5 v_5^{m-1} + \alpha_6 v_6^{m-1}$$

$$v_2^m = \alpha_1 v_3^{m-1} + \alpha_2 v_4^{m-1} + \alpha_3 v_5^{m-1} + \alpha_4 v_6^{m-1} + \alpha_5 v_7^{m-1} + \alpha_6 v_8^{m-1}$$

$$\vdots$$

$$v_{N/2^m}^m = \alpha_1 v_{N/2^{m-1}-1}^{m-1} + \alpha_2 v_{N/2^{m-1}}^{m-1} + \alpha_3 v_1^{m-1} + \alpha_4 v_2^{m-1} + \alpha_5 v_3^{m-1} + \alpha_6 v_4^{m-1}$$

Ecuación 30

$$w_1^m = \beta_1 w_1^{m-1} + \beta_2 w_2^{m-1} + \beta_3 w_3^{m-1} + \beta_4 w_4^{m-1} + \beta_5 w_5^{m-1} + \beta_6 w_6^{m-1}$$

$$w_2^m = \beta_1 w_3^{m-1} + \beta_2 w_4^{m-1} + \beta_3 w_5^{m-1} + \beta_4 w_6^{m-1} + \beta_5 w_7^m + \beta_6 w_8^m$$

$$\vdots$$

$$v_{N/2^m}^m = \beta_1 w_{N/2^{m-1}-1}^{m-1} + \beta_2 w_{N/2^{m-1}}^{m-1} + \beta_3 w_1^{m-1} + \beta_4 w_2^{m-1} + \beta_5 w_3^{m-1} + \beta_6 w_4^{m-1}$$

Ecuación 31

- Obtención de los nuevos coeficientes

De nuevo se vuelve a repetir el algoritmo de cálculo, de modo que este punto queda simplificado a la descripción de apartados anteriores.

3.2.3.3 Reconstrucción de la señal a partir de los coeficientes (primera aproximación)

Se usan los coeficientes de menor a mayor nivel, hasta llegar a una expresión matricial dada la ecuación 32:

$$\begin{bmatrix} f_1 \\ \vdots \\ f_N \end{bmatrix} = \begin{bmatrix} v_1^m \\ \vdots \\ v_{N/2}^m \\ w_1^m \\ \vdots \\ w_{N/2}^m \\ \vdots \\ w_1^1 \\ \vdots \\ w_{N/2}^1 \end{bmatrix}^T x \begin{bmatrix} a^m \\ d^m \\ \vdots \\ d^1 \end{bmatrix}$$

Ecuación 32

De nuevo se muestran los resultados obtenidos, mediantes la figura17:

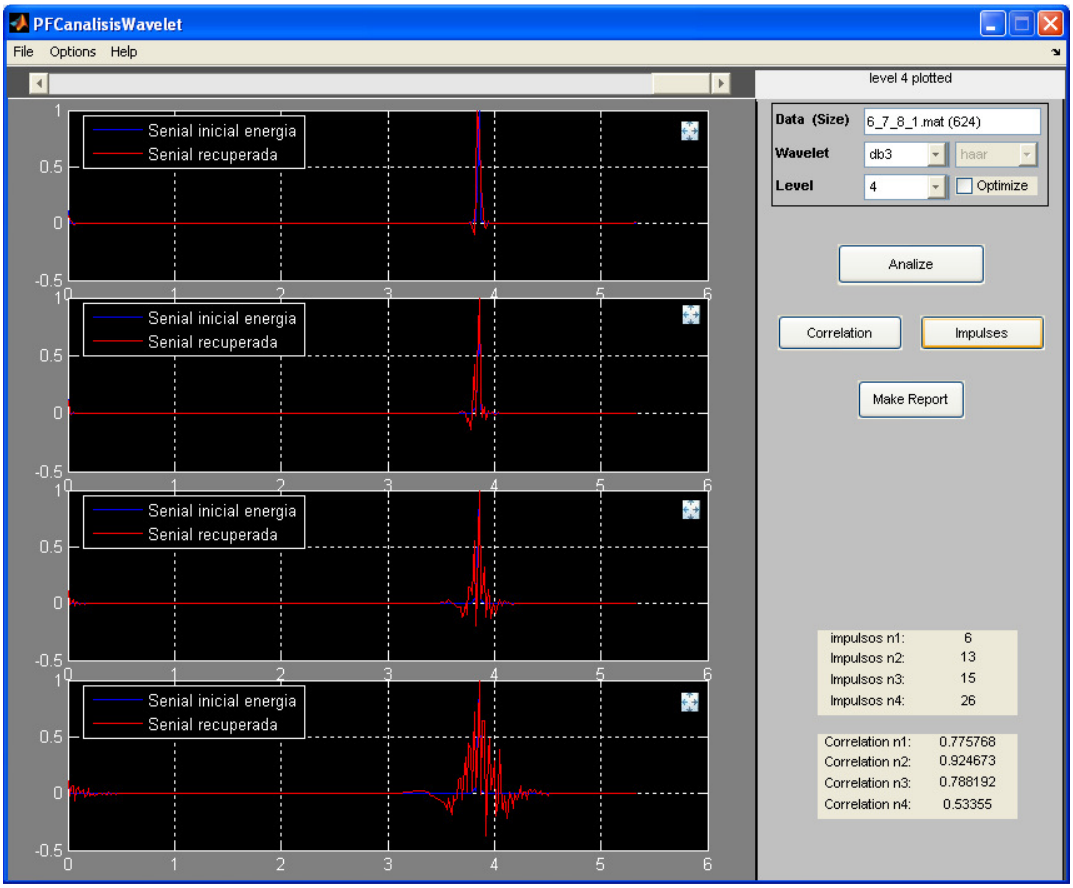


Ilustración 17. Análisis mediante daubechies dB3

En este caso se tiene por primera vez, correlaciones más altas para niveles superiores a uno, sin embargo, la detección correcta sigue siendo la del primer nivel.

Resulta difícil explicar el por qué se tiene mayor correlación para los niveles dos y tres, siendo este el único caso, se clasifica como dato atípico y se pospone su análisis (capitulo 4)

Tanto la anterior transformada como esta, serán en principio, desechadas para la selección de una wavelet madre que mejores parámetros de análisis facilite.

Se pasa al análisis de la señal 6_7_8_2.mat, a la espera de obtener unas correlaciones más elocuentes, y poder comprobar si existe una correcta detección.

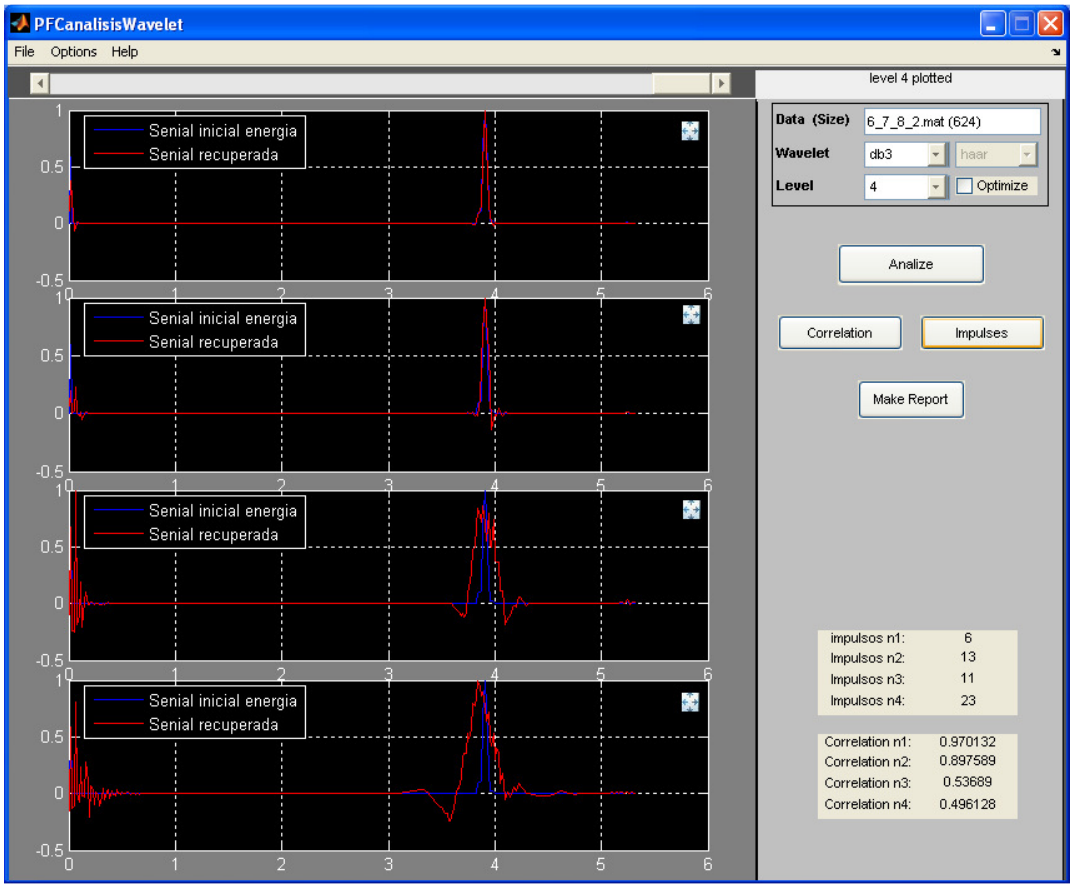


Ilustración 18. Análisis mediante daubehchies dB3

Como era de esperar, se obtiene un valor de correlación lógico entre las señales recuperada e inicial. De nuevo el nivel uno es el único que permite una correcta detección, a priori.

3.2.4. Transformada de coiflets

3.2.4.1. Descripción de la transformada

Siguiendo la sugerencia de Coifman, este tipo de wavelet ortogonales fue construida por Daubechies. El objetivo era conseguir una aproximación mejorada entre los valores de las tendencias y los de la señal original.

3.2.4.2. Obtención de coeficientes: Desarrollo matricial

Tal como se hizo en el apartado anterior, en primer lugar se seleccionan los parámetros adecuados al tipo de wavelet madre.

Selección de los parámetros a usar

Necesarios para la matriz de cálculo de coeficientes de nivel uno y el desplazamiento para niveles sucesivos.

En este caso los parámetros a usar son:

$$\alpha_1 = \frac{1-\sqrt{7}}{16\sqrt{2}}$$
$$\alpha_2 = \frac{5-\sqrt{7}}{16\sqrt{2}}$$
$$\alpha_3 = \frac{14+2\sqrt{7}}{16\sqrt{2}}$$
$$\alpha_4 = \frac{14-2\sqrt{7}}{16\sqrt{2}}$$
$$\alpha_5 = \frac{1-\sqrt{7}}{16\sqrt{2}}$$
$$\alpha_6 = \frac{-3+\sqrt{7}}{16\sqrt{2}}$$

$$\alpha = [\alpha_1 \quad \alpha_2 \quad \alpha_3 \quad \alpha_4 \quad \alpha_5 \quad \alpha_6]$$

Siendo la relación entre los números y los scaling igual que db3:

$$\beta = [\alpha_6 \quad -\alpha_5 \quad \alpha_4 \quad -\alpha_3 \quad \alpha_2 \quad -\alpha_1]$$

$$Desplazamiento = 2$$

Construcción de la matriz transformación de nivel uno

Existen pocas diferencias con respecto a la formación del apartado anterior, tan solo cabe diferenciar los dos últimos casos en vez de tan sólo el último, de forma que la matriz de primer viene dada por la ecuación 33:

$$v_1^1 = (\alpha_3, \alpha_4, \alpha_5, \alpha_6, 0, \ldots 0, \alpha_1, \alpha_2)$$
$$v_2^1 = (\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, 0, \ldots 0)$$
$$v_3^1 = (0, 0, \alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, 0, \ldots, 0)$$
$$\vdots$$

$$v_{N/2}^{1} = (\alpha_5, \alpha_{6,}, 0, ..., 0, \alpha_1, \alpha_2, \alpha_{3,} \alpha_4)$$

Ecuación 33

Y se sigue el mismo desarrollo para w, tenemos la ecuación 34 (calculo matriz en diferencias)

$$w_1^1 = (\beta_{3,}\beta_4, \beta_5\beta_6, 0, 0, ...0, \beta_1, \beta_2)$$
$$w_2^1 = (\beta_1, \beta_{2,}\beta_{3,}\beta_4, \beta_5, \beta_{6,}\, 0, ...0)$$
$$w_3^1 = (0, 0, \beta_1, \beta_{2,}\beta_{3,}\beta_4, \beta_5, \beta_{6,}\, 0, ..., 0)$$
$$\vdots$$
$$w_{N/2}^1 = (\beta_5, \beta_{6,}, 0, ..., 0, \beta_1, \beta_2, \beta_{3,}\beta_4)$$

Ecuación 34

- Calculo de coeficientes y reconstrucción de la señal de nivel uno

Una vez hallada la matriz el cálculo de coeficientes se opera según la ecuación 35:

$$a_m = f \cdot v_m^1 \quad m = 1, ..., N/2. \qquad d_m = f \cdot w_m^1 \quad m = 1, ..., N/2.$$

Ecuación 35

Luego por el momento se tiene:

$$a^1 = (a_1, a_2, ..., a_{N/2}) \qquad d^1 = (d_1, d_2, ..., d_{N/2})$$

Y por tanto ya podemos hacer la primera reconstrucción de la señal como se describió en el desarrollo matricial del apartado 3.1.1.1. según la ecuación 36

$$f_{reconstruida} = \begin{bmatrix} v_1^1 \\ \vdots \\ v_{N/2}^1 \\ w_1^1 \\ \vdots \\ v_{N/2}^1 \end{bmatrix}^T \times \begin{bmatrix} a^1 \\ b^1 \end{bmatrix}$$

Ecuación 36

Esta es la señal con la cual, se irá trabajando para comprobar usando los coeficientes de dicho nivel, se puede recuperar la señal, detectando los impulsos de forma correcta.

- Calculo de la matriz del siguiente nivel

Tanto para este caso como para los siguientes actuaremos de forma recurrente para el cálculo de las matrices de niveles mayores que uno.

De esta para hallar las siguientes matrices nos apoyaremos operando con la de nivel anterior, de esta forma tenemos los vectores dados por la ecuación 37 y 38:

$$v_1^m = \alpha_1 v_1^{m-1} + \alpha_2 v_2^{m-1} + \alpha_3 v_3^{m-1} + \alpha_4 v_4^{m-1} + \alpha_5 v_5^{m-1} + \alpha_6 v_6^{m-1}$$

$$v_2^m = \alpha_1 v_3^{m-1} + \alpha_2 v_4^{m-1} + \alpha_3 v_5^{m-1} + \alpha_4 v_6^{m-1} + \alpha_5 v_7^{m-1} + \alpha_6 v_8^{m-1}$$

$$\vdots$$

$$v_{N/2^m}^m = \alpha_1 v_{N/2^{m-1}-1}^{m-1} + \alpha_2 v_{N/2^{m-1}}^{m-1} + \alpha_3 v_1^{m-1} + \alpha_4 v_2^{m-1} + \alpha_5 v_3^{m-1} + \alpha_6 v_4^{m-1}$$

Ecuación 37

$$w_1^m = \beta_1 w_1^{m-1} + \beta_2 w_2^{m-1} + \beta_3 w_3^{m-1} + \beta_4 w_4^{m-1} + \beta_5 w_5^{m-1} + \beta_6 w_6^{m-1}$$

$$w_2^m = \beta_1 w_3^{m-1} + \beta_2 w_4^{m-1} + \beta_3 w_5^{m-1} + \beta_4 w_6^{m-1} + \beta_5 w_7^m + \beta_6 w_8^m$$

$$\vdots$$

$$v_{N/2^m}^m = \beta_1 w_{N/2^{m-1}-1}^{m-1} + \beta_2 w_{N/2^{m-1}}^{m-1} + \beta_3 w_1^{m-1} + \beta_4 w_2^{m-1} + \beta_5 w_3^{m-1} + \beta_6 w_4^{m-1}$$

Ecuación 38

- Obtención de los nuevos coeficientes

De nuevo se vuelve a repetir el algoritmo de cálculo, de modo que este punto queda simplificado a la descripción de apartados anteriores.

3.2.4.3. Reconstrucción de la señal a partir de los coeficientes (primera aproximación)

Se usan los coeficientes de menor a mayor nivel, hasta llegar a una expresión matricial dada por la ecuación 39:

$$\begin{bmatrix} f_1 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ f_N \end{bmatrix} = \begin{bmatrix} v_1^m \\ \vdots \\ v_{N/2}^m \\ w_1^m \\ \vdots \\ w_{N/2}^m \\ \vdots \\ w_1^1 \\ \vdots \\ w_{N/2}^1 \end{bmatrix}^T x \begin{bmatrix} a^m \\ d^m \\ \vdots \\ d^1 \end{bmatrix}$$

Ecuación 39

De nuevo se muestran los resultados obtenidos en la ilustración19:

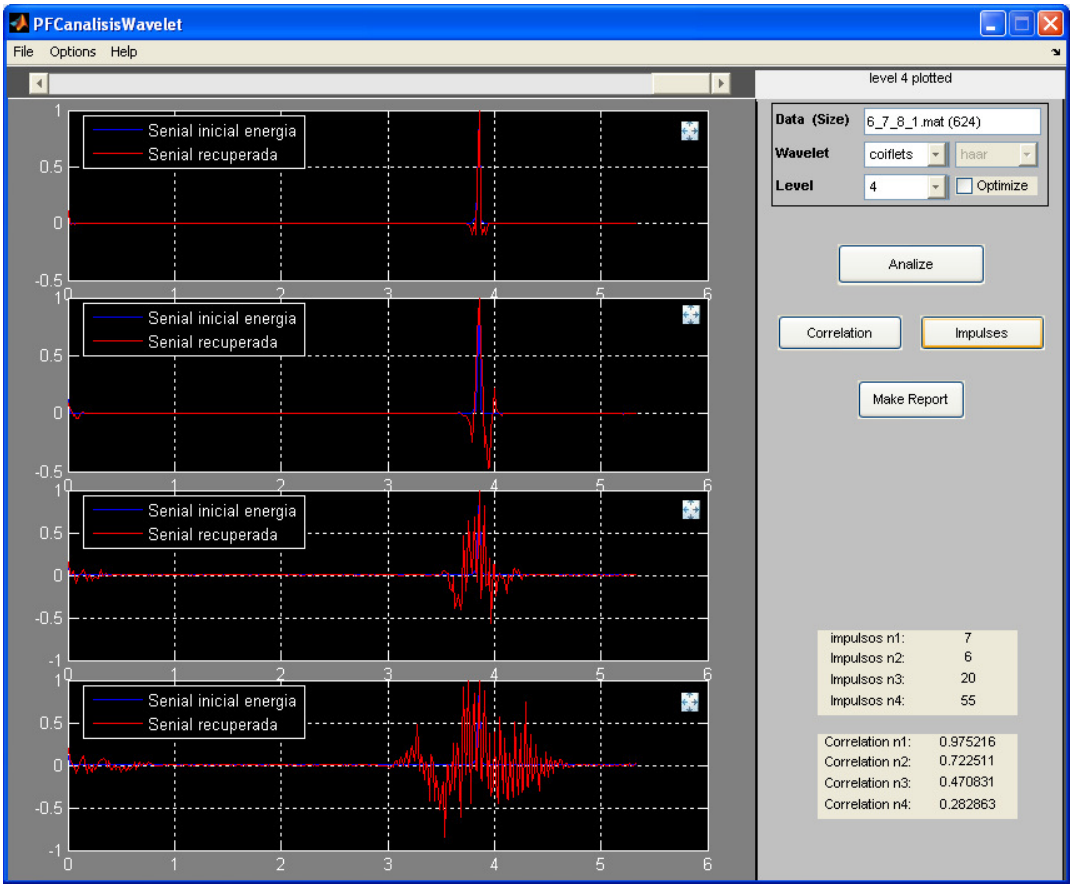


Ilustración 19. Análisis mediante la transformada de coilflets

La correlación es muy baja y la detección no permite la diferenciación de los 6impulsos de la señal en el primer nivel. Sin analizar aún el tipo de

transformada, se observan que este tipo de filtro, inserta vibraciones altas en frecuencia, lo cual distorsiona la detección de los impulsos y hace sospechar que la detección para el nivel 2 es susceptible de error.

Pasando de nuevo a analizar la señal 6_7_8_2.mat tenemos la ilustración20:

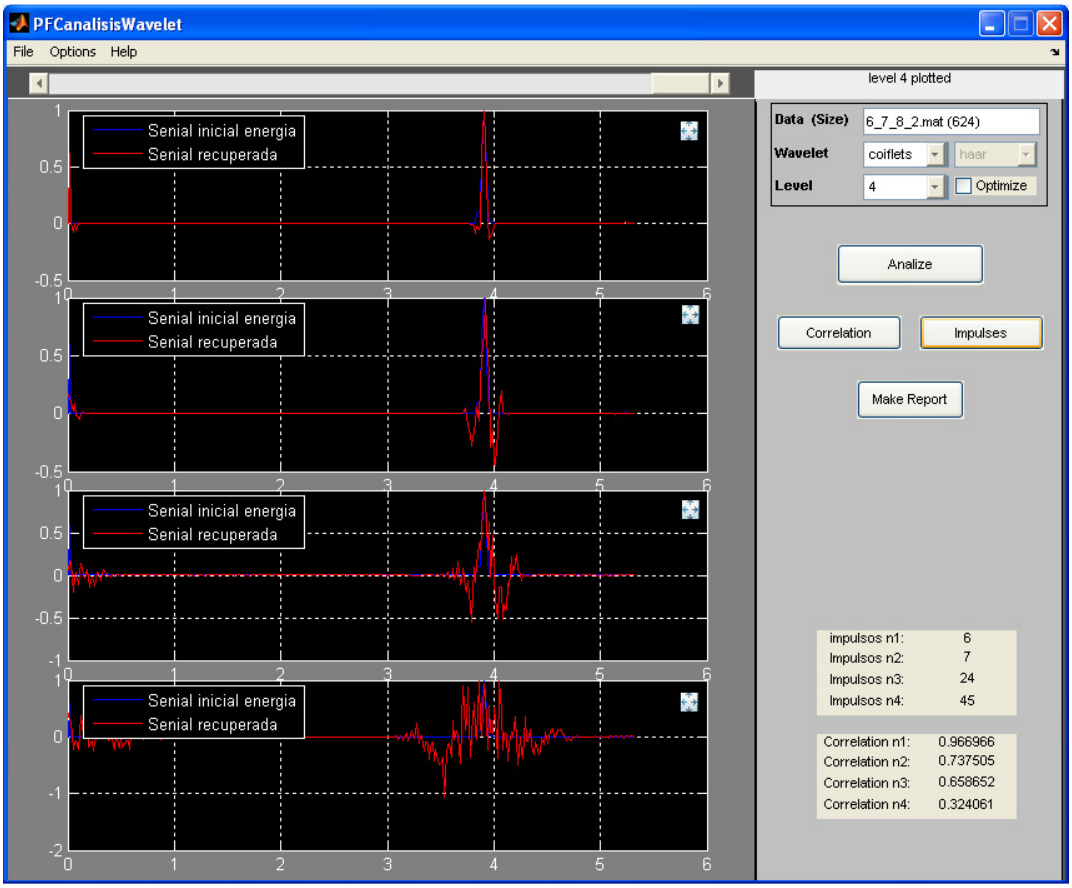


Ilustración 20. Análisis mediante la transformada de Coiflets

Idéntico resultado, lo que muestra la reticencia con este primer análisis a usar este tipo de transformada.

3.2.5. Análisis mediante las funciones wavelet de Matlab.

3.2.5.1. Descripción y propósito. (Base teórica)

Previo análisis de los impulsos obtenidos, y con objetivo de comprobar la fiabilidad de los métodos implementados, pasamos a contrastar los datos implementados por funciones matlab, así como transformadas no usadas, para corroborar que el método que se seleccione como wavelet madre, es el idóneo.

Sin entrar en detalles de cómo han sido implementadas las funciones de matlab se observa un primer caso, en el que las funciones han sido usadas de igual forma que las desarrolladas mediante algoritmos matriciales.

Para este caso se añadió una pestaña más en la interfaz a modo de que el usuario pudiera seleccionar las funciones matlab[3], para así habilitar los análisis pertinentes con estas funciones, de este modo se tienen las cuatro transformadas analizadas anteriormente, a los que se les ha añadido el la transformada biorthogonal.

Los tipos analizados fueron:

- T. Haar
- T. Daubechies dB2
- T. Daubechies dB3
- T. Coiflets
- T. Bior

3.2.5.2. Resultados (Análisis de la señal en energía)

Se clasifican los resultados de cada una de las transformadas, sin embargo, se hace un análisis conjunto de los mismos por motivos obvios, descrito al final de los mismos.

[3] Para más detalles consultar especificaciones de uso de la interfaz

1. Haar

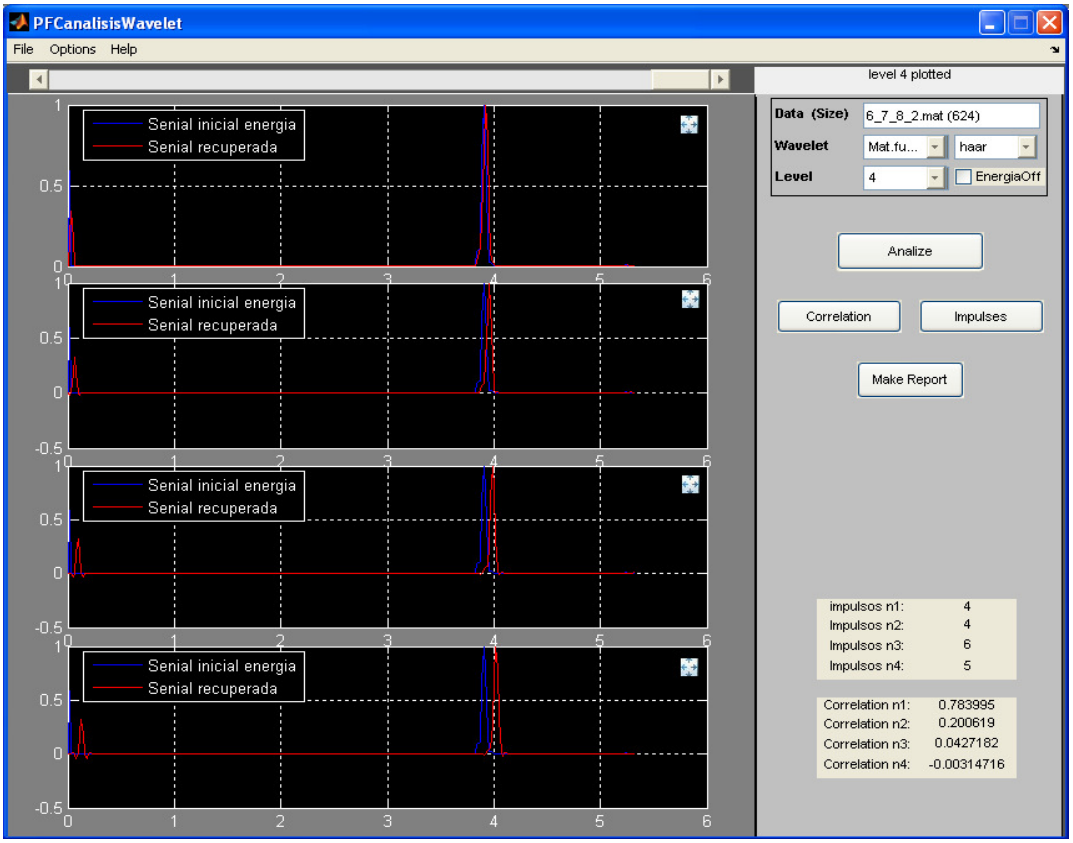


Ilustración 21. Análisis mediante funciones matlab en energía (Haar)

Ampliación del nivel 1, mostrado en la ilustración22

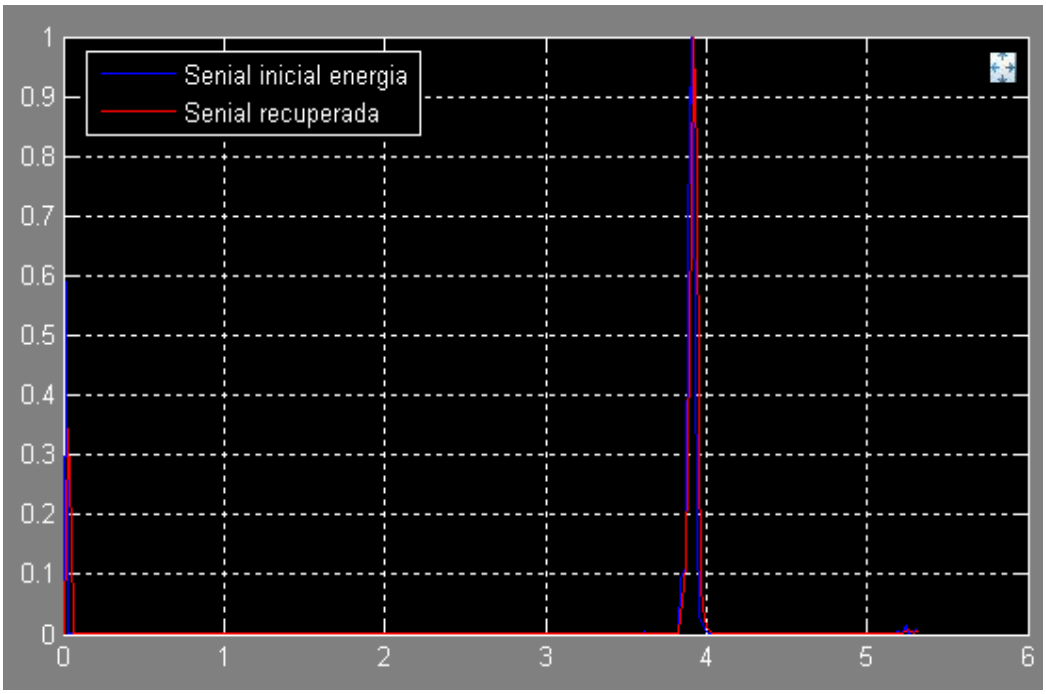


Ilustración 22. Ampliación del nivel 1

2. Daubechies db2, ilustraciónnúmero 23:

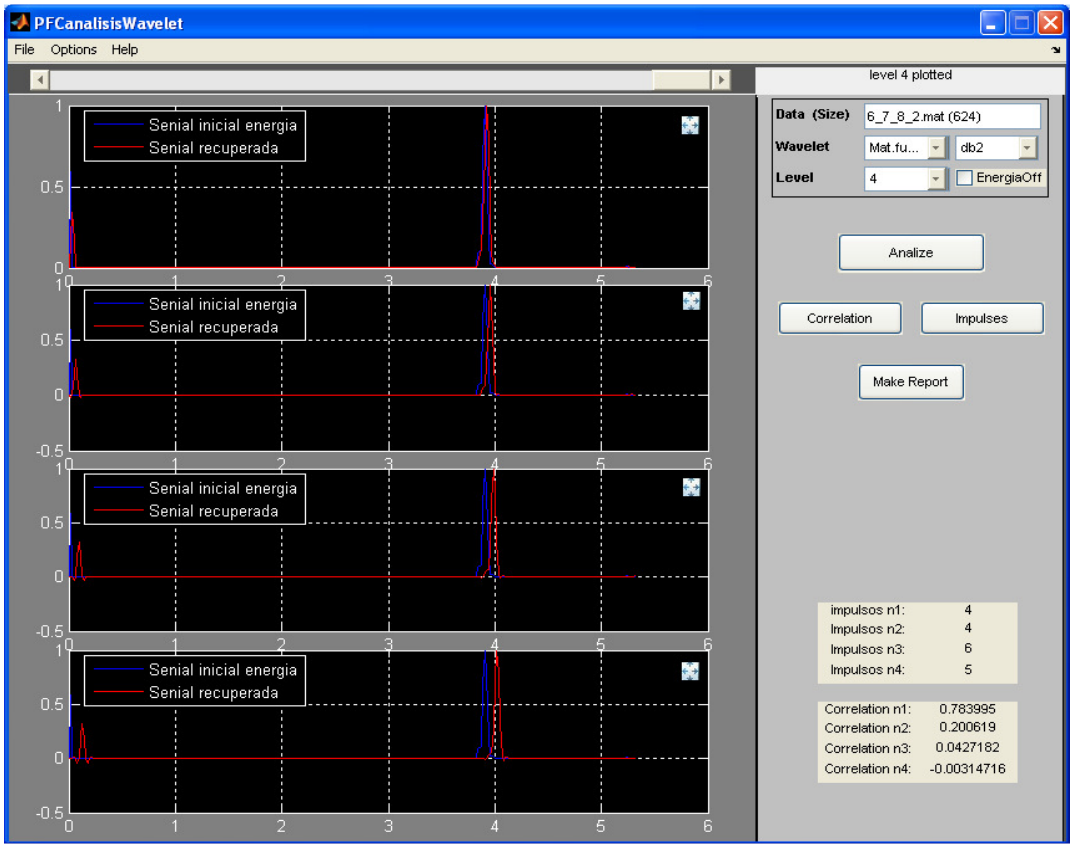


Ilustración 23. Análisis mediante funciones matlab en energía (Daubechies dB2)

Ampliación nivel 1, ilustraciónnúmero 24:

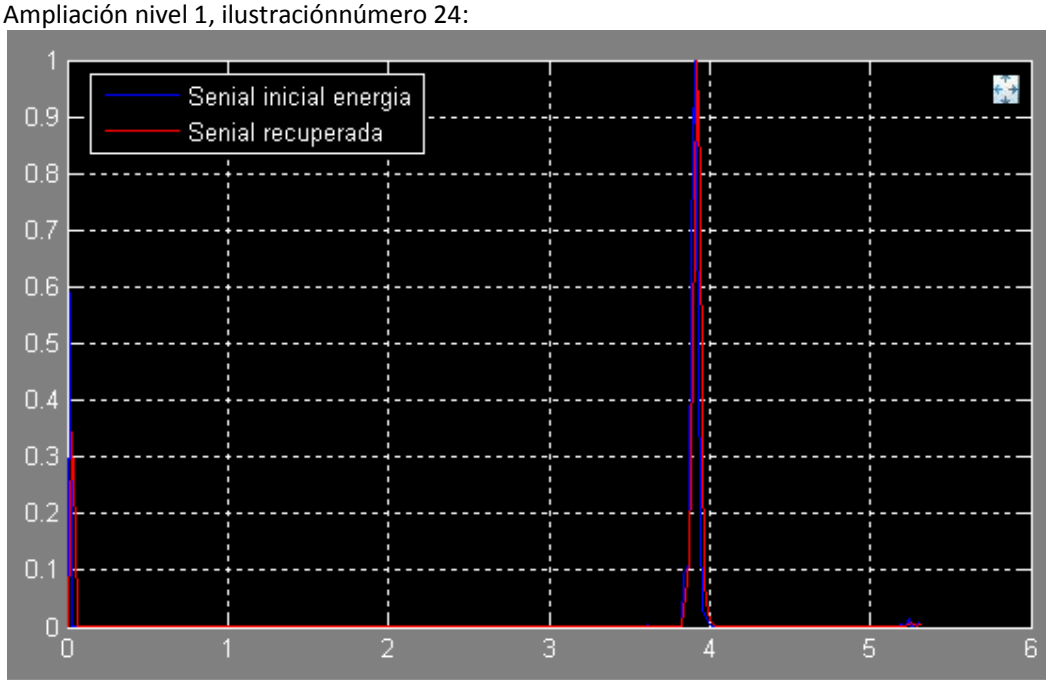


Ilustración 24. Ampliacion del nivel 1

3. Daubechies nivel3, ilustración25:

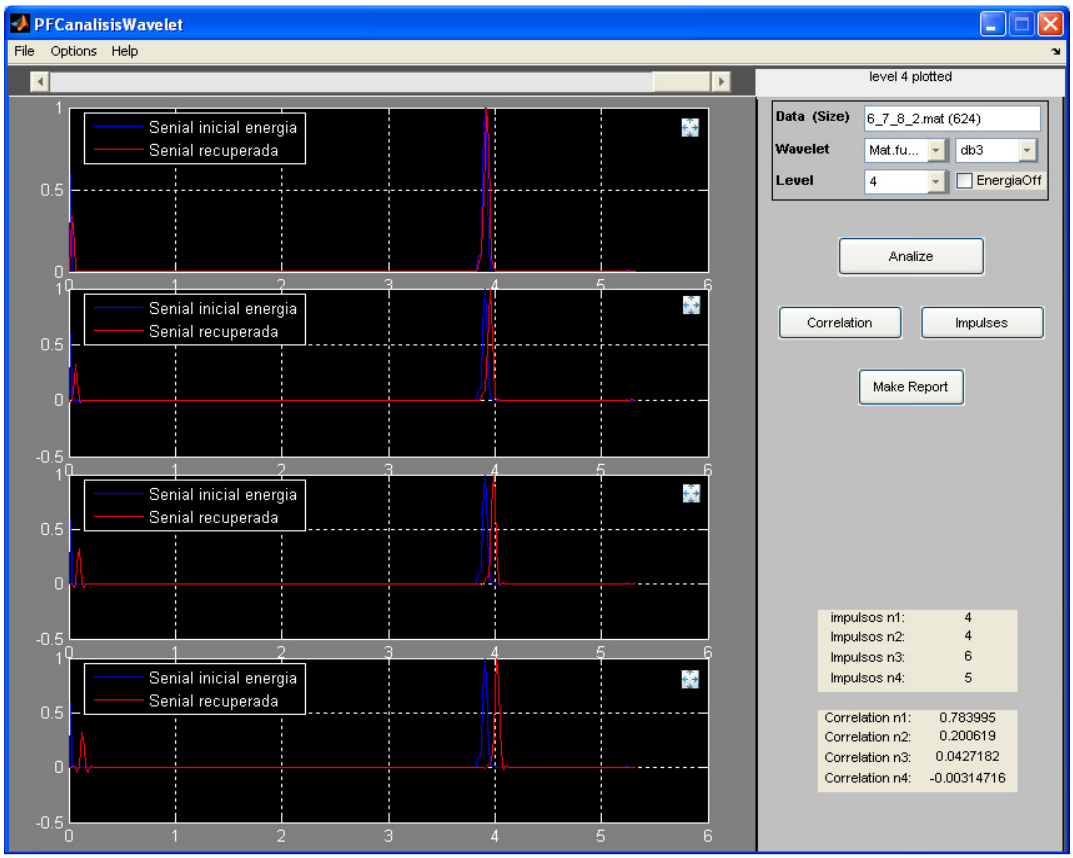


Ilustración 25. Análisis mediante funciones matlab en energía (Daubechies dB3)

Ampliación nivel 1, ilustración26:

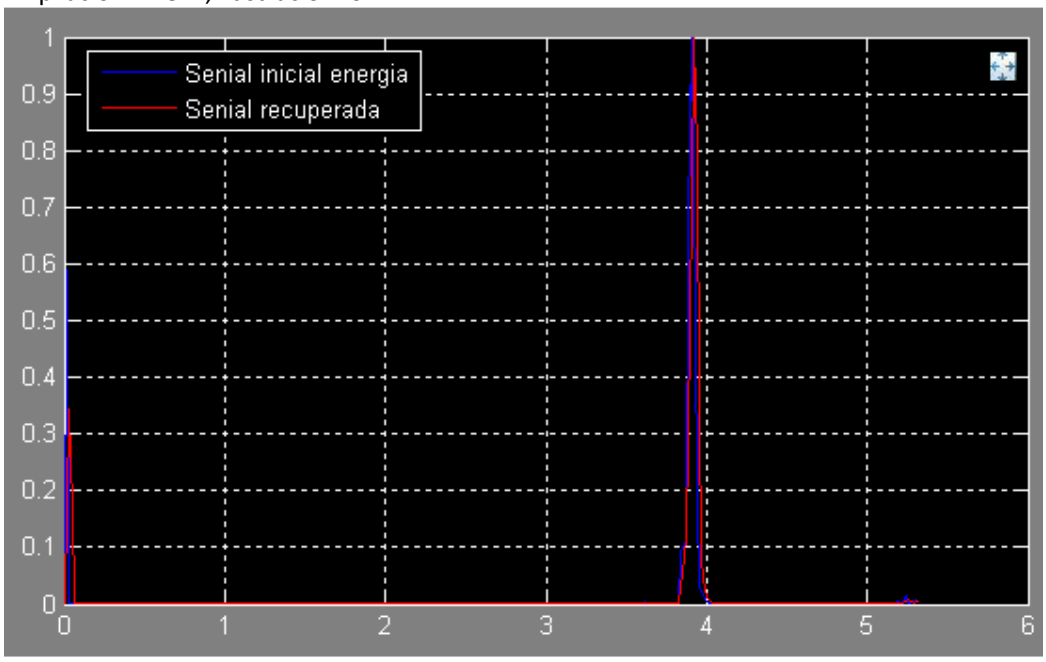


Ilustración 26. Ampliación de la señal de nivel 1

4. Coiflets, ilustraciónnúmero 27:

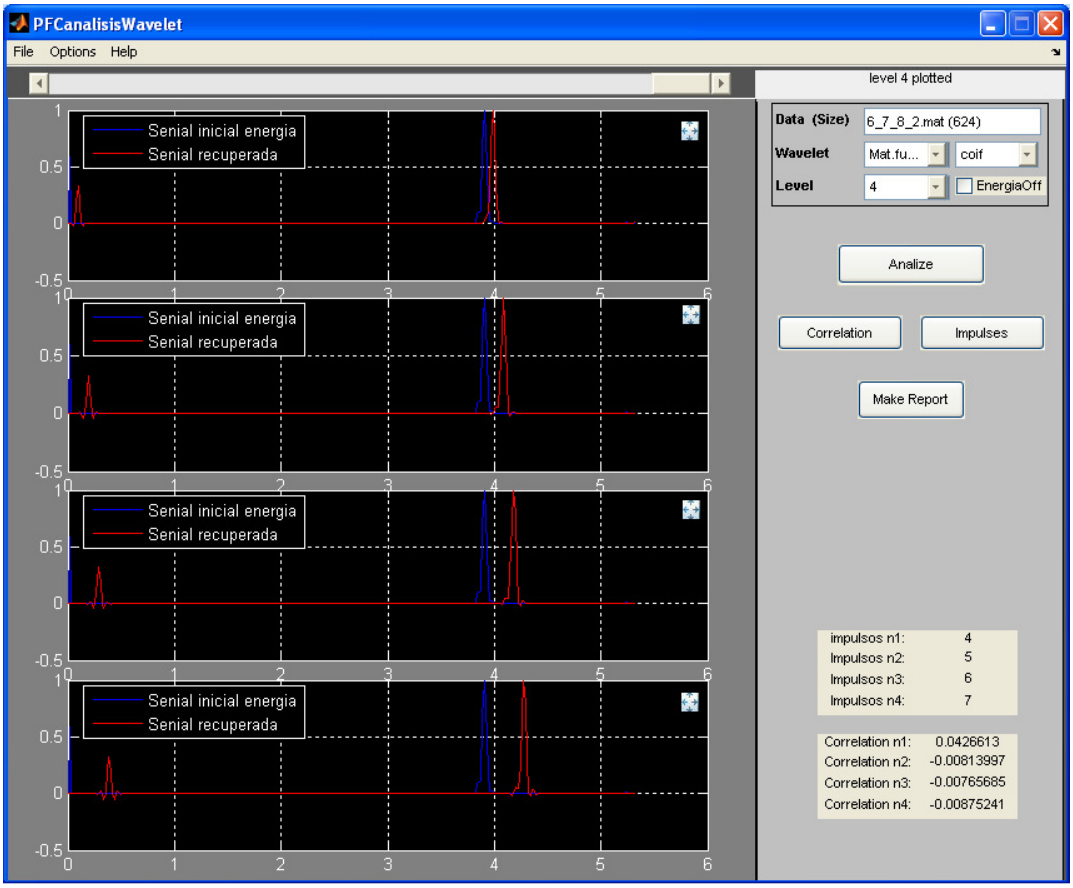


Ilustración 27. Análisis mediante funciones matlab en energía (Coiflets)

Ampliación nivel1, ilustraciónnúmero 28:

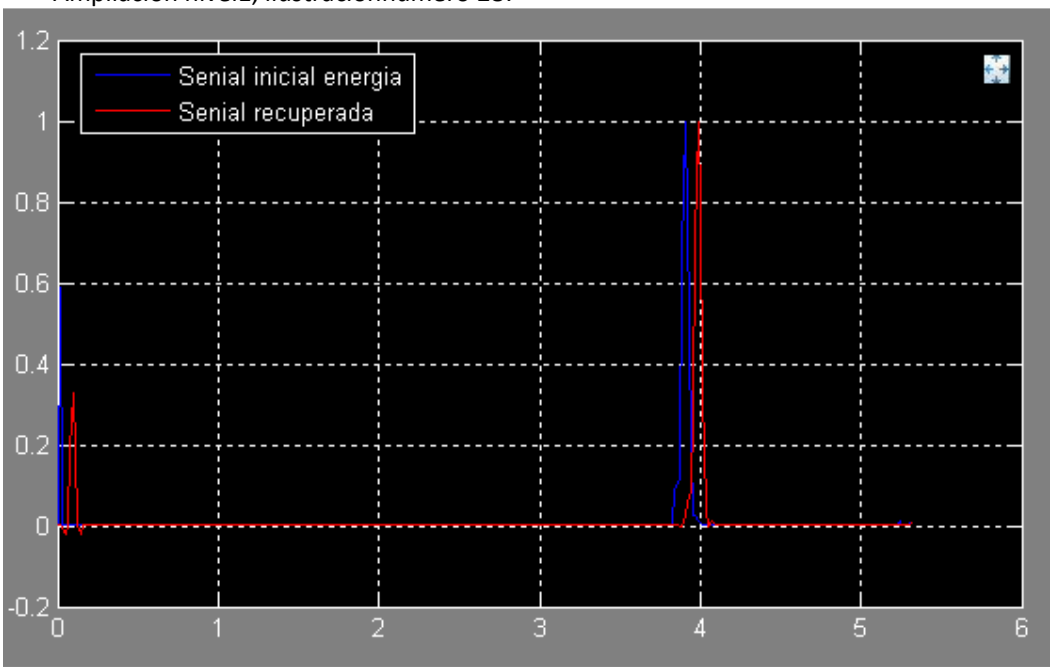


Ilustración 28. Ampliación del nivel 1

5. Biorthogonal

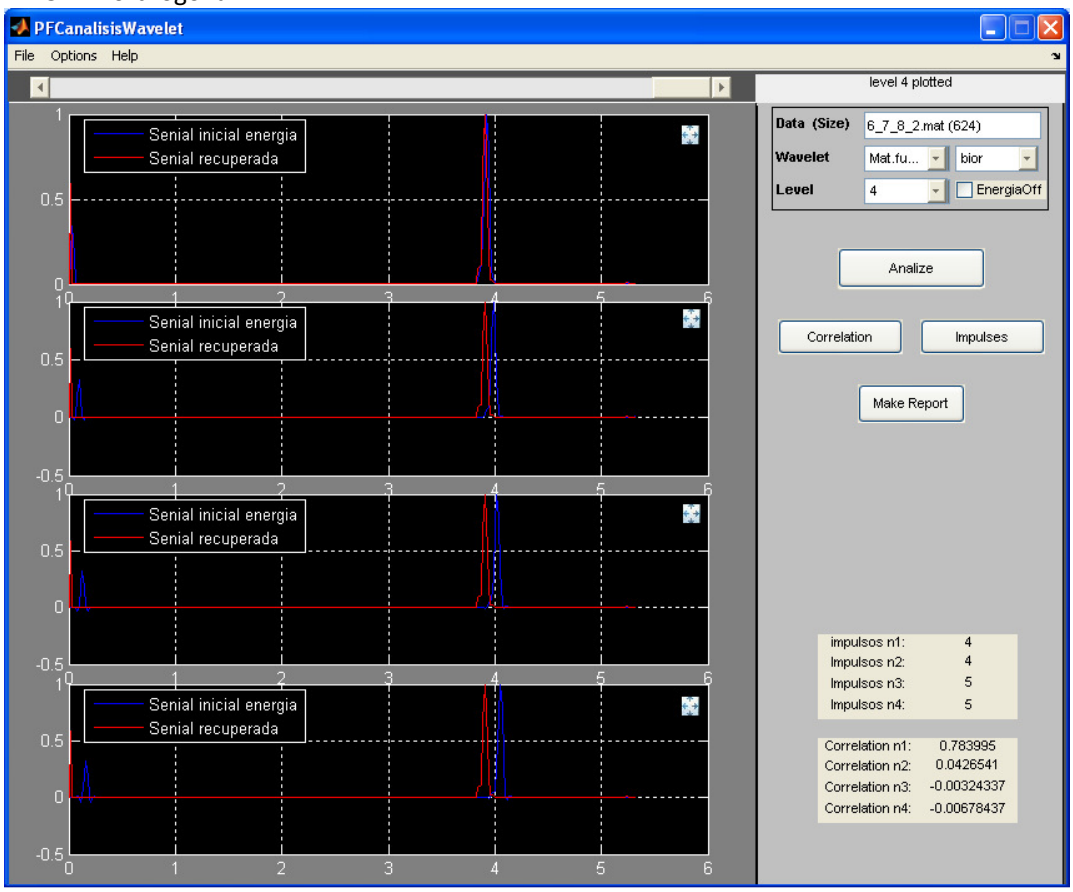


Ilustración 29. Análisis mediante funciones matlab en energía (Biorthogonal)

Ampliación nivel 1, mostrada en la ilustración30.

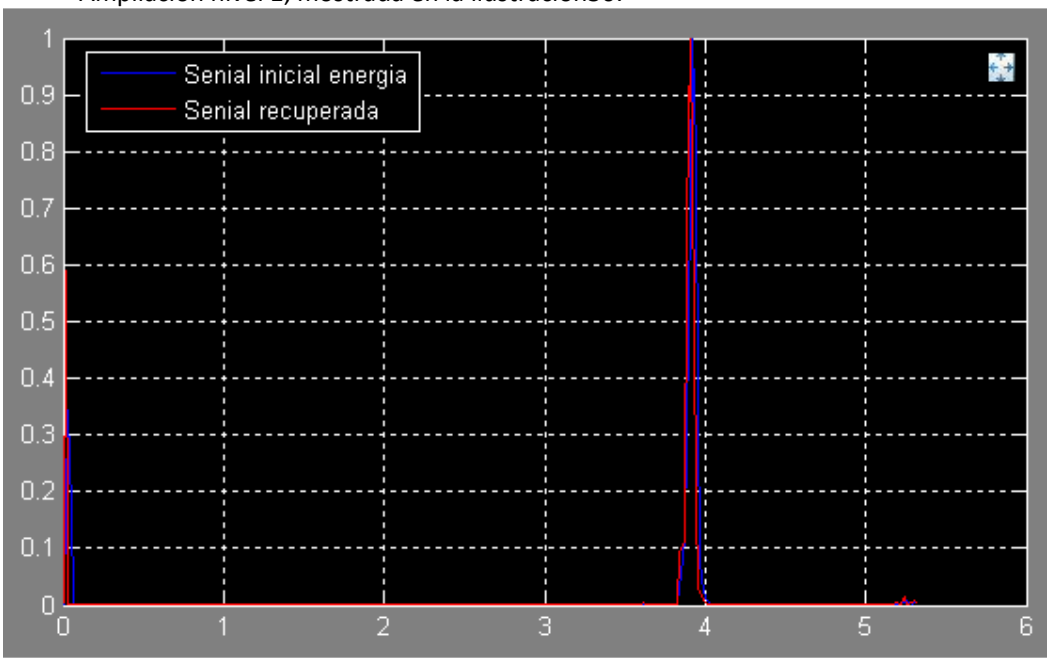


Ilustración 30. Ampliación del nivel 1

Como se observa de las gráficas anteriores, la detección de los impulsos bajo las mismas condiciones, comparando las funciones propias de matlab, con las funciones desarrolladas (usando método matriciales), deja un balance muy favorable a favor de estas últimas.

La causa principal se debe a que con las funciones propias de matlab el número de muestras que usamos es muy pequeño, reseñar que si se desea usar la herramienta wavemenu, de matlab, no es posible, dado que el número de muestras es muy reducido.

Es por este motivo, que se pasó a hacer el análisis de la señal por medio de funciones propias de matlab, pero usando todas las muestras.

Para esto se habilita una nueva opción, con la cual se usa la señal tal cual, y no la señal en energía, como se había venido haciendo hasta entonces.

3.2.5.3. Resultados (Análisis de la señal en tiempo)

Este nuevo análisis, será propio de las funciones de matlab, ya que como se comentó, no es posible realizar este tipo de análisis con el desarrollo matricial debido a la finita capacidad de cálculo de matlab.

Con este tipo de análisis tenemos de nuevo la posibilidad de hacer una selección de coeficientes. Dado que realizar otro estudio sobre las funciones de matlab no está dentro de los límites de este proyecto, se hará un análisis básico.[4] (Representada en este momento la señal reconstruida tanto con coeficientes de aproximación y diferencias)

Con este método se pretende poder establecer una comparación de resultados obtenidos mediantes los diversos caminos, así como tener otra vía para alcanzarlos.

Evidentemente, tendrá sus ventajas e inconvenientes, lo cual se desgranará en el apartado final, discusión global de los resultados.

Tan solo reseñar que para poder comparar los métodos, y detectar los impulsos, tras el análisis, se toma la energía de la señal.

Una vez dicho esto, se pasa a la presentación de los resultados, para todos los tipos habilitados, al igual que se hizo en apartados sucesivos.

Haar:

[4] Los detalles relacionados con el análisis de la señal temporal con las funciones de Matlab se analizaran en el apartado donde se comentan los resultados obtenidos.

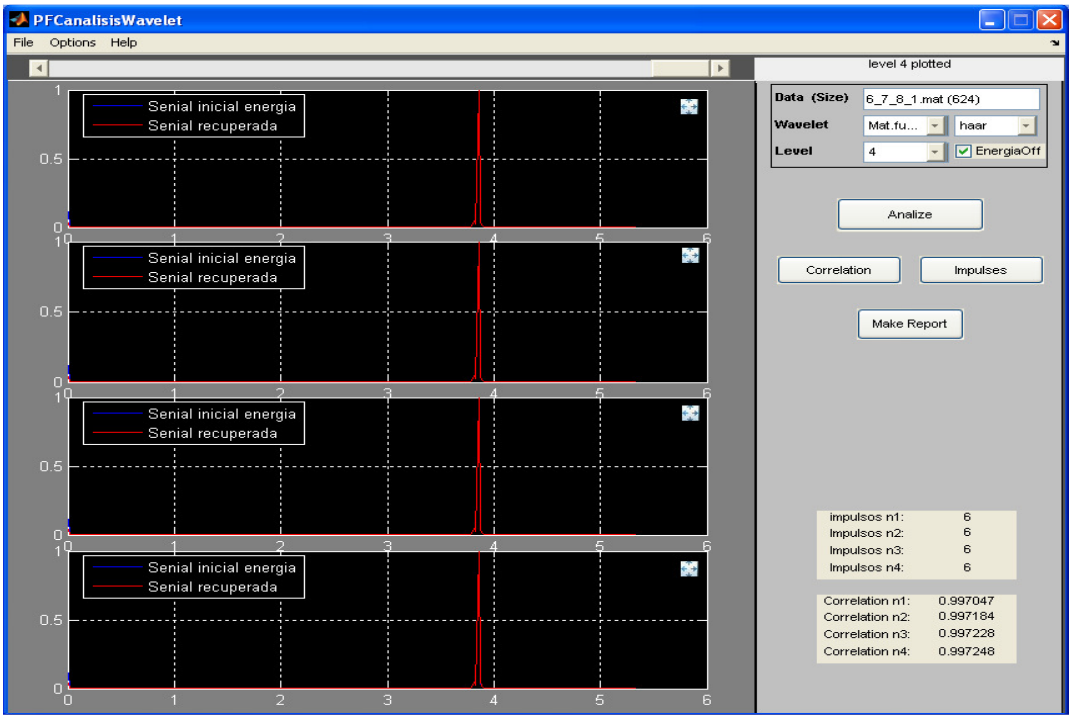


Ilustración 31. Análisis mediante funciones matlab, señal en tiempo (Haar)

Dada la gran correlación que existe en este tipo de análisis, se obvia la señal ampliada de aproximación de primer nivel. Si se muestra sin embargo el análisis de la señal 6_7_9_1.mat en la ilustración32.

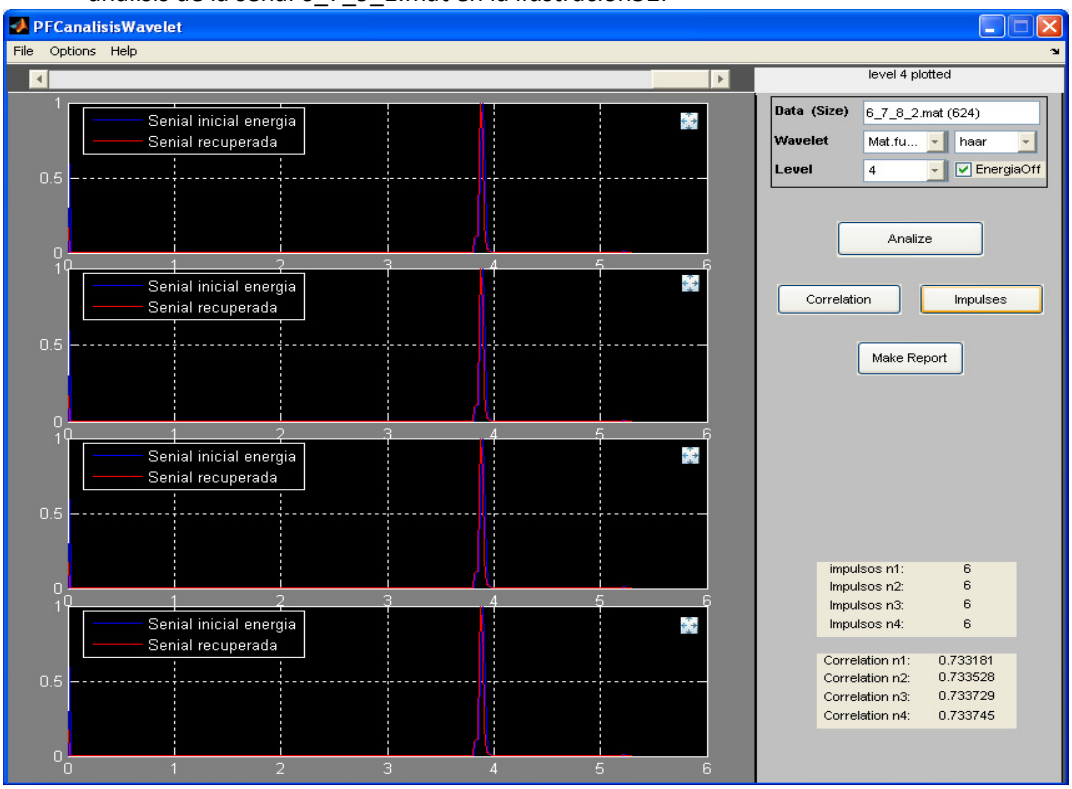


Ilustración 32. Análisis mediante funciones Matlab, señal en tiempo (Haar 2)

Daubechies dB2, ilustración33:

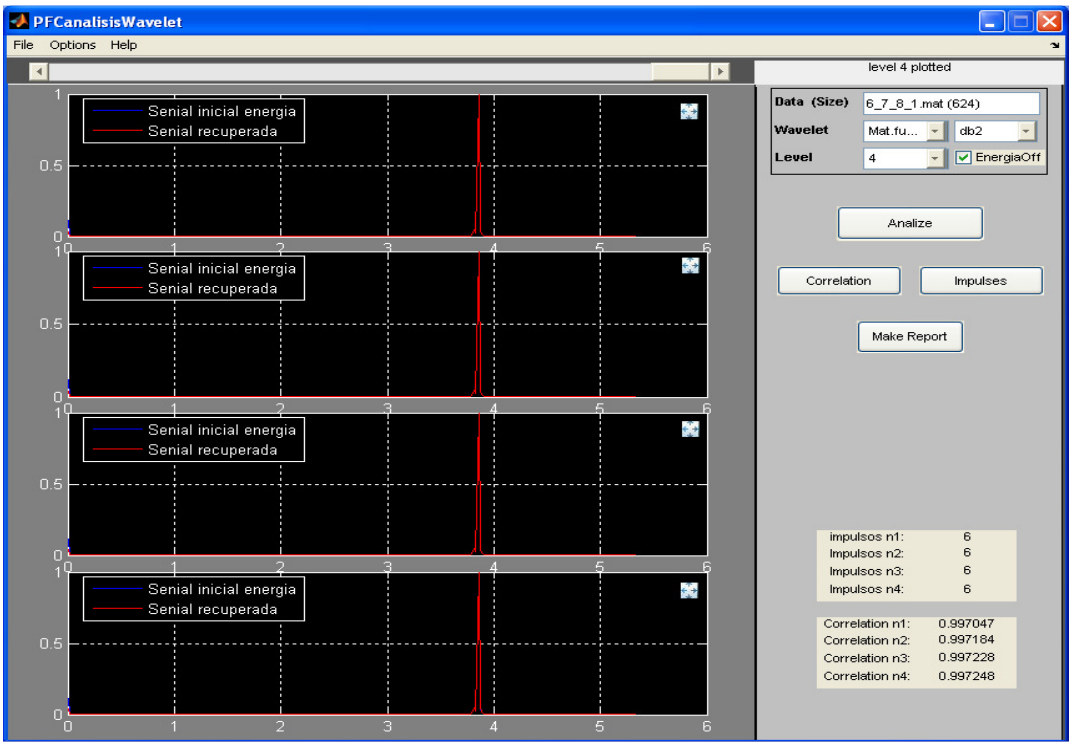


Ilustración 33. Análisis mediante funciones Matlab, señal en tiempo (Daubechies db2)

Señal 6_7_8_2.mat, ilustración34:

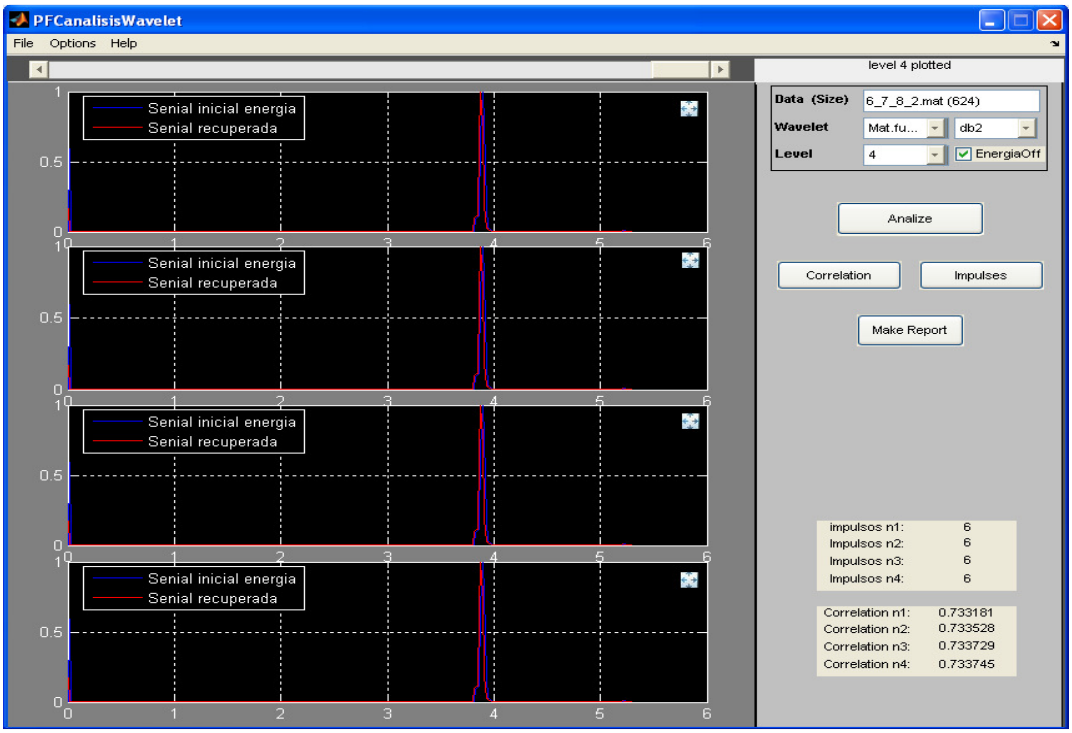


Ilustración 34. Análisis mediante funciones Matlab, señal en tiempo (Daubechies dB2 2)

Daubechies dB3, ilustración35:

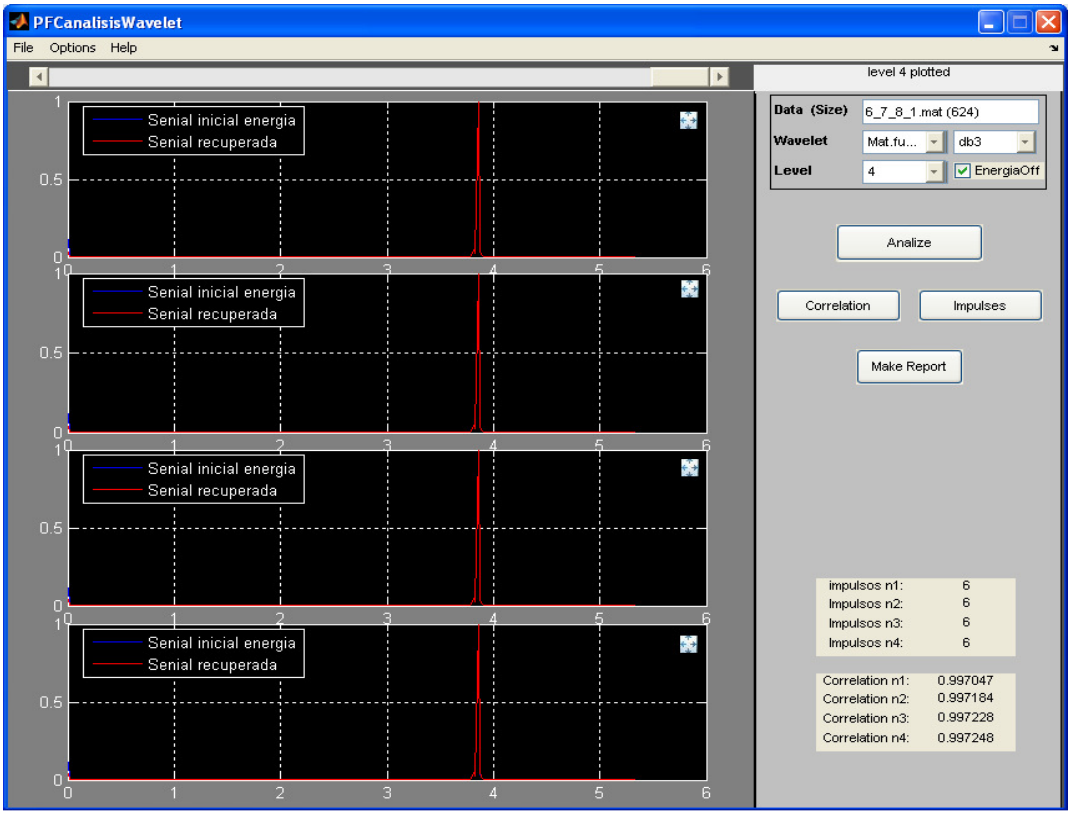


Ilustración 35Análisis mediante funciones Matlab, señal en tiempo (daubechies dB3)

Señal 6_7_8_2.mat, ilustración36:

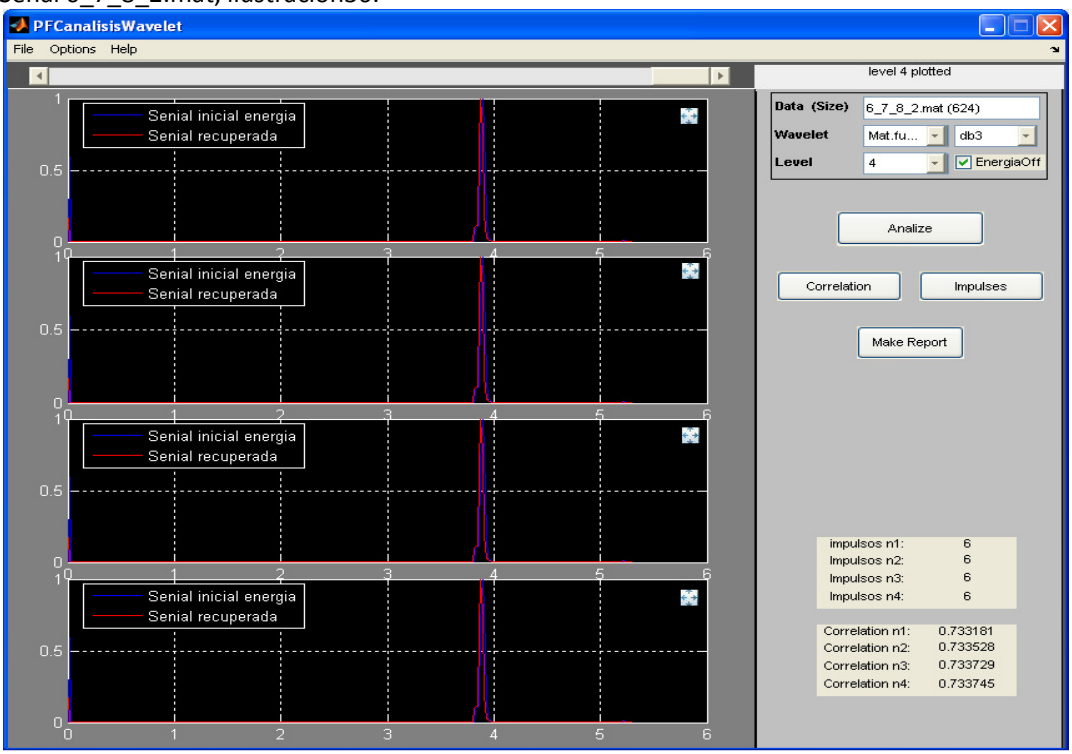


Ilustración 36. Análisis mediante funciones Matlab, señal en tiempo (Daubechies dB3 2)

Coiflets, ilustración36:

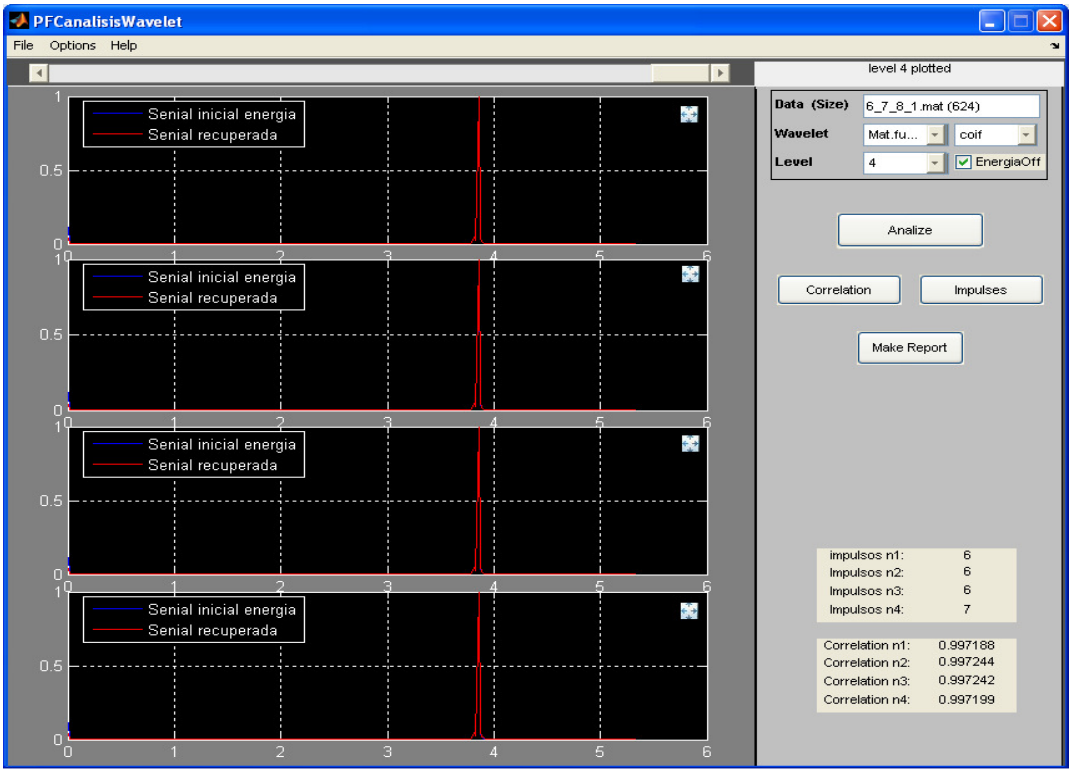


Ilustración 37. Análisis mediante funciones Matlab, señal en tiempo (Coiflets)

Análisis de la señal 6_7_8_2.mat:

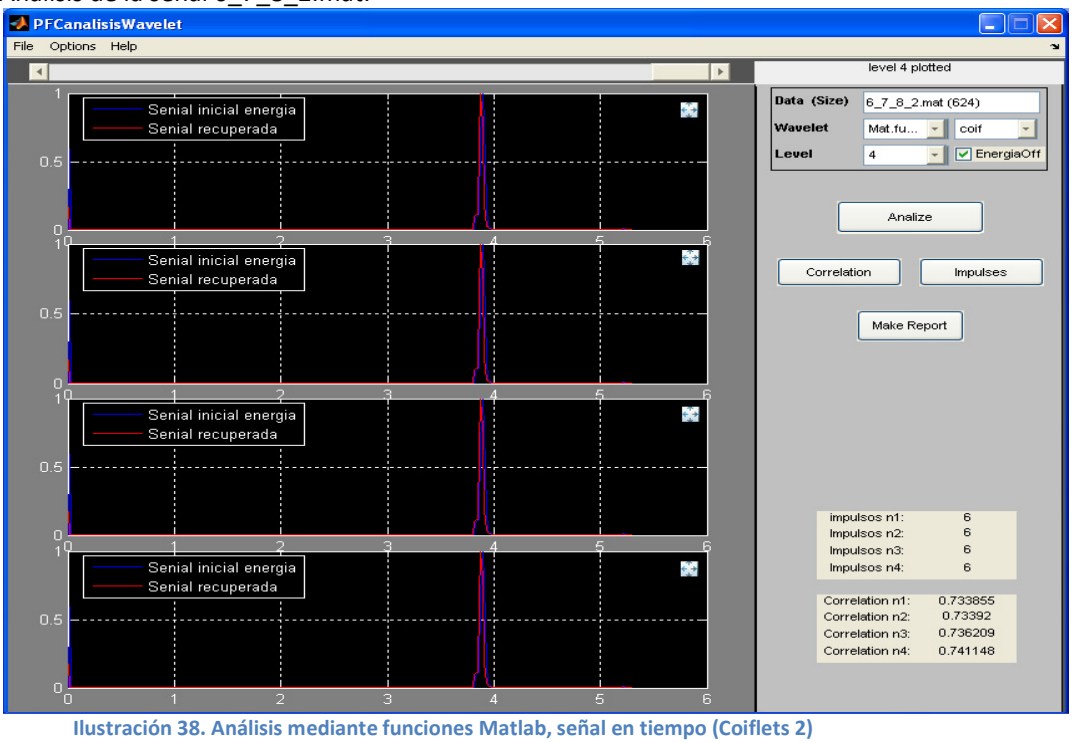


Ilustración 38. Análisis mediante funciones Matlab, señal en tiempo (Coiflets 2)

Biorthogonal, ilustración38:

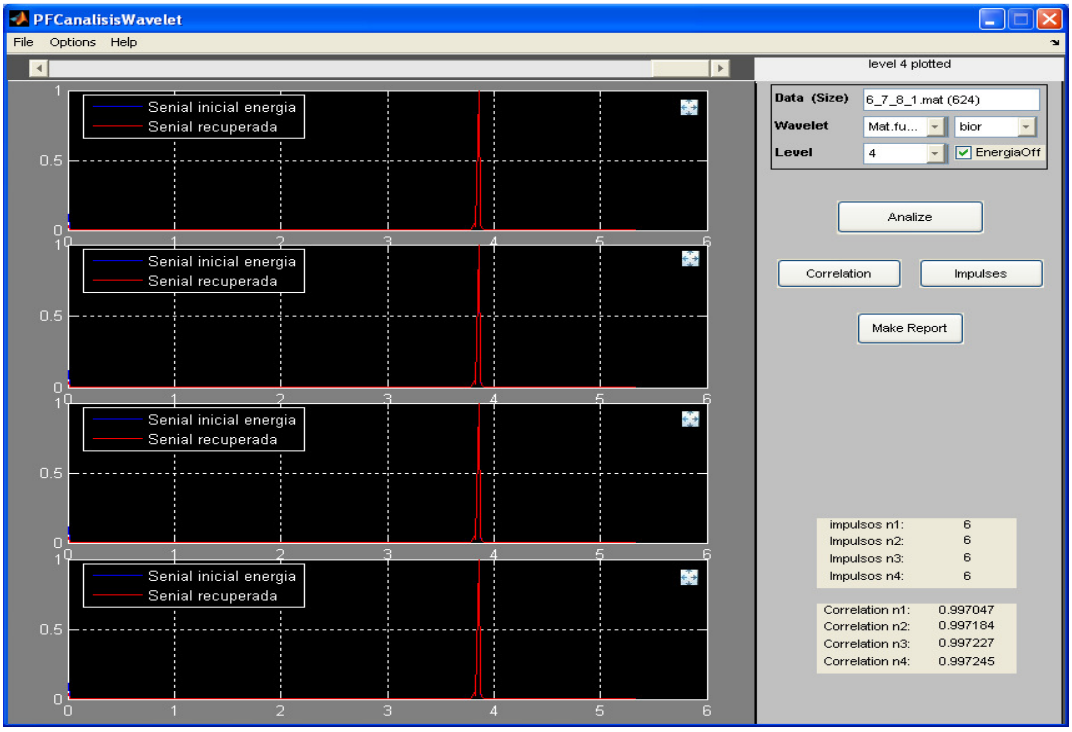


Ilustración 39. Análisis mediante funciones Matlab, señal en tiempo (Biorthogonal)

Análisis de la señal 6_7_8_2.mat:

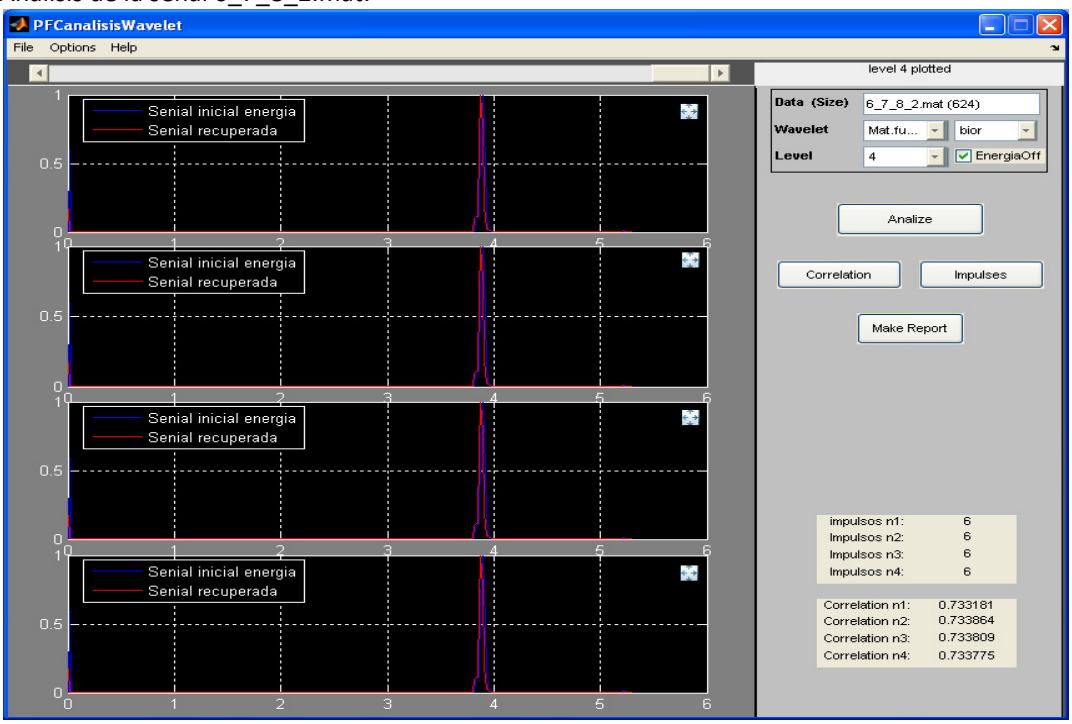


Ilustración 40. Análisis mediante funciones Matlab, señal en tiempo (Biorthogonal 2)

CAPITULO 4. ANÁLISIS Y COMPARACIÓN DE RESULTADOS, CONCLUSIÓNES.

Tras la presentación de los distintos métodos de análisis, se van a analizar los resultados de los mismos, tomando como ejemplo las señales 6_7_8_1.mat y 6_7_8_2.mat.

Como se comentó en el apartado 3, el método que hace uso de las funciones de matlab, y la señal en energía, queda descartado, por no conseguirse con este tipo de análisis, la detección correcta de los impulsos.

Dicho esto se tiene, en primer lugar, el análisis del método matricial, para la señal 6_7_8_1.mat y posteriormente para la señal 6_7_8_2.mat.

En segundo lugar se muestra el método que hace uso de las funciones matlab, de nuevo con las dos señales: 6_7_8_1.mat y 6_7_8_2.mat.

Y en tercer y último lugar, se muestra una comparación entre los dos métodos para cada una de las señales. (Se seleccionan los parámetros de diagnostico mas apropiados, para comparar ambos métodos).

4.1. RESULTADOS.

En todas las tablas, se tiene en cada fila, la wavelet madre seleccionada en cada caso. Y en cada columna, el nivel analizado.

Dentro de cada wavelet madre y cada nivel, se destaca, en una primera tabla (tabla 1), el número de impulsos detectados, y la correlación entre la señal recuperada con dicha transformada y la señal de entrada.

1. Métodos matriciales

	Nivel1		Nivel2		Nivel3		Nivel4	
	Impulsos	**Corr**	**Impulsos**	**Corr**	**Impulsos**	**Corr**	**Impulsos**	**Corr**
Haar	**6**	**1**	5	0.740208	**6**	**0.704912**	4	0.46164
dB2	**6**	**0.887617**	12	0.769091	12	0.61905	**6**	**0.375871**
dB3	**6**	**0.776139**	13	0.925459	15	0.789129	26	0.53235
Coiflets	7	0.97558	**6**	**0.722968**	20	0.471216	55	0.2828
6_7_8_1.mat								

Tabla 1. Impulsos y correlaciones, métodos matriciales para la señal 6_7_8_1.mat

Es en el nivel 1 donde el método matricial es capaz de detectar la totalidad de los impulsos (excepto el tipo Coiflets).

Otro resultado destacable es, que la señal recuperada con la wavelet madre de Haar posee una correlacion igual a 1 respecto a la señal inicial en energía, con lo que no ofrece duda alguna de la exactitud en la detección.

Llama la atención, que sea sin embargo con el tipo dB3, con el que se obtienen las correlaciones mas elevadas para los restantes niveles. Cada una de las wavelet madre, se diferencia en la forma del filtro que se emplea y eso hará, que conjunto a la forma de la señal, la correlación o la detección de los impulsos varíen de unas a otras.

Cómo el objetivo es, la correcta detección de los impulsos, independientemente del tipo, o de la señal que se tenga a la entrada, el estudio de los efectos que pueda tener cada transformada con respecto a la señal de entrada queda fuera de los márgenes de estudio de la herramienta de análisis desarrollada.

Por tanto, dado que lo que se desea es, la extracción de los parámetros de diagnóstico, se comprueba si los impulsos detectados son validos. Para saberlo, se muestran en la tabla 2, las características de los mismos. (Tiempo entre impulsos en segundos y energía normalizada).

En un segunda tabla (tabla 2), se tiene, cada vez que han sido detectados los 6 impulsos, el tiempo entre ellos (en segundos) y la energía. Una última columna (la de más a la derecha) indica el numero de coeficientes de la transformada de wavelet que son necesarios para recuperar la señal.

Además se explica el significado de la posición de cada uno de los tiempos y de las energías a continuación, para un correcto entendimiento de la tabla 2. (Se toma de ejemplo la wavelet madre de Haar, nivel1)

- Tiempo entre impulsos:

3.5577 → entre el primer y el segundo impulso
0.064103 →entre el segundo y el tercer impulso
0.16026→entre el tercero y el cuarto impulso
0.032051→entre el cuarto y el quinto impulso
1.3301→entre el quinto y el sexto impulso

- Energía de los impulsos:

0.0067616 → energía del primer impulso
7.8903e-006 → energía del segundo impulso
1.5362e-005 → energía del tercer impulso
0.12375→ energía del cuarto impulso
5.8815e-006→ energía del quinto impulso
1.5731e-005→ energía del sexto impulso

Características impulsos	Tiempo entre impulsos	Energía	Numero de coeficientes
Haar(Nivel1)	3.5577 0.064103 0.16026 0.032051 1.3301	0.0067616 7.8903e-006 1.5362e-005 0.12375 5.8815e-006 1.5731e-005	320(aproximación)+ 320(diferencias)
Haar(Nivel3)	3.5577 0.048077 0.17628 0.048077 1.3301	0.034689 1.2426e-005 2.3676e-005 0.46082 2.0345e-005 2.4638e-005	160(aproximación) + 160(detalle)
dB2 (Nivel 1)	3.5257 0.048077 0.14423 0.032051 1.3141	0.0045205 1.1298e-005 2.1109e-005 0.37526 4.7918e-005 8.2324e-005	320(aproximación)
dB3 (Nivel 1)	3.5737 0.12821 0.048077 0.048077 1.3301	0.0015649 9.8694e-006 0.00043754 0.31517 0.00031189 3.8922e-005	320(aproximación)
Coiflets(Nivel 2)	0.096154 3.4776 0.14423 0.096154 1.218	0.0030491 9.6316e-005 5.8598e-005 0.30012 0.010265 1.6429e-005	160(aproximación)
6_7_8_1.mat			

Tabla 2. Características de los impulsos de la señal 6_7_8_1.mat

Vemos como existe gran similitud, entre las detecciones de Haar y dB2, mientras que en dB3 y coiflets los parámetros de diagnóstico difieren en gran medida.

A la hora de plantearse, cuales son los parámetros que se deben tomar para el correcto diagnóstico, resulta evidente coger aquellos que poseen una mayor correlación entre la señal de entrada y la recuperada tras el análisis.

Es por esto que el primer análisis realizado indica que las transformadas de Haar y la transformada Daubechies db2 son las mas apropiadas para la realización del análisis. Como en el primer caso, la correlación vale 1, estos serán los parámetros modelo (válidos para clasificar la señal)

Como se detalla en la columna "Numero de coeficientes", para todos los casos, excepto con la transformada de Haar (nivel3) y Coiflets (nivel2), se han usado todos los coeficientes, luego se han respetado todas las frecuencias de la señal de entrada. Vemos como la transformada de wavelet tiene poco margen de actuación, ya que en los casos donde sí se ha filtrado la señal, la correlación baja.

Se pasa ahora a tomar la señal 6_7_8_2.mat para intentar corroborar lo concluido anteriormente.

	Nivel1		**Nivel2**		**Nivel3**		**Nivel4**	
	Impulsos	**Corr**	**Impulsos**	**Corr**	**Impulsos**	**Corr**	**Impulsos**	**Corr**
Haar	**6**	**1**	4	0.914081	**6**	**0.900054**	4	0.501063
dB2	**6**	**1**	8	0.873763	10	0.590667	6	0.345867
dB3	**6**	**0.970913**	13	0.90299	11	0.549153	23	0.504097
Coiflets	**6**	**0.969033**	7	0.740009	24	0.662085	45	0.326791
6_7_8_2.mat								

Tabla 3. Impulsos y correlaciones, métodos matriciales para la señal 6_7_8_2.mat

De la tabla 3 se pueden sacar las mismas conclusiones que se obtuvieron para la señal 6_7_8_1.mat , ya que son las transformadas de Haar y daubechies dB2, las que presentan mayores correlaciones, y por tanto cabe esperar unos parámetros de diagnóstico idóneos e iguales para ambos casos, ya que la correlación es 1.

Se pasa a clasificar los parámetros de salida en la tabla 4:

Características impulsos	Tiempo entre impulsos	Energía	Numero de coeficientes
Haar (Nivel1)	3.5897 0.22436 1.1859 0.048077 0.048077	0.11619 6.619e-006 0.17677 5.8664e-006 9.5675e-005 2.2063e-005	320(aproximación) + 320(detalle)
Haar (Nivel 3)	3.7981 0.048077 0.048077 1.234 0.048077	0.37766 0.017209 0.54532 6.883e-005 5.6055e-005 1.8653e-005	80 (aproximación) + 80+160(detalle)
dB2(Nivel1)	3.5897 0.22436 1.1859 0.048077 0.048077	0.11619 6.619e-006 0.17677 5.8664e-006 9.5675e-005 2.2063e-005	320(aproximación) + 320 (diferencias)
dB3(Nivel1)	0.048077 3.734 0.032051 1.218 0.032051	0.049997 0.00012614 0.16658 0.00013158 1.5741e-005 1.1362e-005	320(aproximación)
Coiflets(Nivel 1)	3.5897 0.27244 1.25 0.048077 0.048077	0.12766 6.8844e-006 0.37792 5.924e-006 9.2118e-005 1.65e-005	320(aproximación) + 320 (diferencias)
6_7_8_2.mat			

Tabla 4. Características de los impulsos de la señal 6_7_8_2.mat

Dados los resultados, se evidencia la relación, entre correlación de la señal de entrada y la recuperada, y la calidad de los tiempos y energías obtenidos.

Para el caso de la señal 6_7_8_2.mat, tanto con la transformada de wavelet tipo Haar y tipo Daubechies dB2 (donde la correlación fue 1), se tienen unos valores de tiempos y energías iguales, y perfectos para la clasificación posterior.

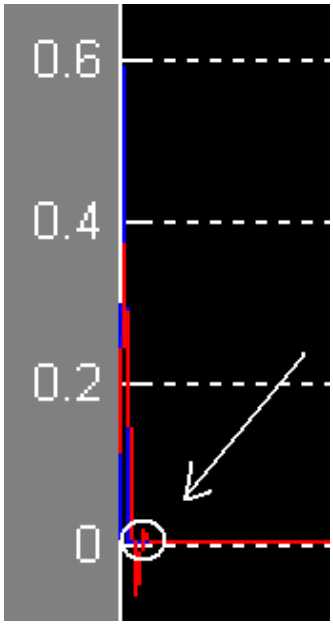


Ilustración 41. Efecto del filtro dB3 en el pulso 1º

Reseñar que en el caso de los resultado con la transformada de wavelet dB3, la correlación también es muy alta (0.970913), lo hace que en un principio, se esperen resultados parecidos a los destacados anteriormente.

Sin embargo, debido al tipo de wavelet se generan impulsos espurios, y se evidencian impulsos que deberían detectarse.

Para ilustrar este hecho, se muestra la ilustración41, donde a continuación del primer pulso, se muestra pegado a él, el impulso espureo que no debería ser detectado, y que perturba el análisis (por esto los tiempos son tan diferentes).

Por último, a lo que se refiere al tipo de datos obtenidos con la wavelet madre de Coiflets, se observa una gran similitud a los parámetros caracterizados con los tipos T. de Haar y dB2.

Esta similitud concuerda tanto en cuanto la correlación de la señal recuperada con respecto a la inicial, es muy elevada, pero inferior a 1.

2. Funciones Matlab (Señal en tiempo).

Una vez analizado el método matricial, viendo el poco margen de maniobra que se deja a la transformada de wavelet y para corroborar los resultados, se realiza este segundo análisis.

Los parámetros analizados son los mismos, sin embargo, ahora la señal analizada no está en energía, si no en tiempo, con lo que posee todas las componentes frecuenciales de la señal de entrada, luego toda vez que para la reconstrucción de la señal se han usado los coeficientes en aproximación, en cada uno de los niveles se hace una descomposición frecuencial mitad.

	Nivel1		Nivel2		Nivel3		Nivel4	
	Impulsos	Corr	Impulsos	Corr	Impulsos	Corr	Impulsos	Corr
Haar	6	0.997047	6	0.997184	6	0.997228	6	0.997248
dB2	6	0.997047	6	0.997184	6	0.997228	6	0.997248
dB3	6	0.997047	6	0.997184	6	0.997228	6	0.997248
Coiflets	6	0.997188	6	0.997244	6	0.997242	7	0.997199
Bior	6	0.997047	6	0.997184	6	0.997227	6	0.997245
6_7_8_1.mat								

Tabla 5. Impulsos y correlaciones, métodos matlab para la señal 6_7_8_1.mat

Características impulsos	Tiempo entre impulsos	Energía	Numero de coeficientes
En todos los casos	3.5577 0.064103 0.16026 0.032051 1.3462	0.0067616 7.8903e-006 1.5362e-005 0.12375 5.8815e-006 1.5802e-005	500000
6_7_8_1.mat			

Tabla 6. Características de los impulsos de la señal 6_7_8_1.mat

Los resultados muestras a priori la idoneidad del método.

En primer lugar, podría sorprender que en todos los casos (excepto Coiflets nivel4) se detecten 6 impulsos con una correlación tan alta entre la señal de entrada y la recuperada, lo que hacer suponer la correcta extracción de los parámetros de salida. Sin embargo, es un resultado esperado:

Dado que la información de la señal que interesa, se encuentra a frecuencias muy bajas (como queda demostrado cuando se hace la señal en energía con 640 muestras y aún así se detectan bien los impulsos), la transformada de wavelet, no afecta a la detección en sus primeros niveles, ya que se sigue manteniendo toda la información, en las frecuencias que se mantienen en la señal recuperada.

Es por esto, que como el cálculo de la señal en energía posterior, tampoco afecta a la no detección de los impulsos, se obtienen unos parámetros aceptables para todos los casos.

Al tener un apartado a propósito para comparar ambos métodos (matricial y func. Matlab), es en este punto donde se extraen las conclusiones finales.

Por último, se presentan los resultados alcanzados con la señal 6_7_8_2.mat en las tablas 7 y 8, esperando ratificar el razonamiento anterior.

	Nivel1		Nivel2		Nivel3		Nivel4	
	Impulsos	Corr	Impulsos	Corr	Impulsos	Corr	Impulsos	Corr
Haar	6	0.733181	6	0.733528	6	0.733729	6	0.73375
dB2	6	0.733181	6	0.733528	6	0.733729	6	0.73375
dB3	6	0.733181	6	0.733528	6	0.733729	6	0.73375
Coiflets	6	0.733855	6	0.73392	6	0.736209	6	0.741148
Bior	6	0.73318	6	0.733864	6	0.733809	6	0.733775
6_7_8_2.mat								

Tabla 7. Impulsos y correlaciones, métodos matlab para la señal 6_7_8_2.mat

En este caso, sorprende obtener una correlación, al menos, dos decimas inferior a la analizada con respecto a la señal 6_7_8_1.mat.

Sin embargo, mediantes el estudio de la señal, se tiene, que en este caso particular, la detección del primer pulso se retrasa.

La causa por la que se ha producido este efecto, es el filtrado que se produce con la transformada de wavelet en la señal en tiempo.

El filtrado elimina vibraciones espureas que acompañan al primer impulso a tratar, de esta forma, la detección en este caso se produce justo cuando comienza verdaderamente el primer impulso, mientras que en el caso del análisis matricial, se adelanta el impulso espureo, tomandose ambos (espureo y no espureo) como uno sólo en la detección con la señal en energía.

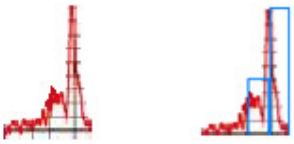


Ilustración 42. Simulación impulso 1 sin filtrado

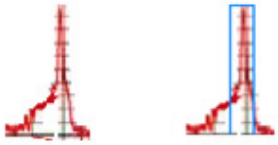


Ilustración 43. Simulación impulso 1 con filtrado (T.Wavelet)

Luego el análisis concreto de esta señal, aporta una nueva visión del problema, relegando el método matricial a un segundo plano, y proponiendo este análisis como línea a seguir. El razonamiento final y comparaciones entre resultados que veremos en el apartado posterior serán concluyentes.

Se presentan en la tabla 8 los parámetros que se obtienen con esta señal una vez detectados los impulsos.

Características impulsos	Tiempo entre impulsos	Energía	Numero de coeficientes
Haar, dB2,dB3,Coiflets,Bior.	3.5789 0.22368 1.1823 0.047932 0.047932	0.17428 6.5912e-006 0.17677 5.5832e-006 9.5017e-005 2.1593e-005	
Nivel1			500000
Nivel2			
Nivel3			
Nivel4			
6_7_8_2.mat			

Tabla 8. Características de los impulsos para la señal 6_7_8_2.mat

A priori los datos de la tabla 8, resultan elocuentes a la espera de la comparación con el método matricial.

4.2. COMPARATIVA DE RESULTADOS ENTRE AMBOS MÉTODOS.

Finalmente se puede comparar las características de los impulsos obtenidos con cada método tomando aquellos donde se ha tenido una mayor correlación entre la señal de entrada y la señal recuperada. En caso de tener varios métodos con la misma correlación, se ha seleccionada la transformada de wavelet tipo Haar, ya que en general, ha proporcionado mejores resultados.

Para la señal 6_7_8_1.mat, tenemos los resultado representados en la tabla 9:

Comparativa	Tiempo entre impulsos	Energía de impulsos
Met. Matriciales	3.5577 0.064103 0.16026 0.032051 1.3301	0.0067616 7.8903e-006 1.5362e-005 0.12375 5.8815e-006 1.5731e-005
Met. Funciones matlab	3.5577 0.064103 0.16026 0.032051 1.3462	0.0067616 7.8903e-006 1.5362e-005 0.12375 5.8815e-006 1.5802e-005
6_7_8_1.mat		

Tabla 9. Comparativa para la señal 6_7_8_1.mat

Ambos método nos aportan idénticos resultados, luego en este caso, si se tuviera que seleccionar alguno de ellos para la realización del análisis, se tomaría el método matricial, dado que los coeficientes que permiten la recuperación de la señal son mucho menores y tenemos una mayor transparencia con dicho método, no siendo necesario un análisis frecuencial de la señal.

Para la señal 6_7_8_2.mat, se tiene la tabla 10:

Comparativa	Tiempo entre impulsos	Energía de impulsos
Met. Matriciales	3.5897 0.22436 1.1859 0.048077 0.048077	0.11619 6.619e-006 0.17677 5.8664e-006 9.5675e-005 2.2063e-005
Met. Funciones matlab	3.5789 0.22368 1.1823 0.047932 0.047932	0.17428 6.5912e-006 0.17677 5.5832e-006 9.5017e-005 2.1593e-005
6_7_8_2.mat		

Tabla 10. Comparativa para la señal 6_7_8_2.mat

Con el análisis de esta señal, vemos la principal y única diferencia existente entre ambos análisis.

La energía del primer impulso es la culpable de que ambos métodos difieran, lo que significa que se debe elegir uno de los dos.

Según el correcto razonamiento del apartado anterior y visto en las figuras 42 y 43, el método de análisis realizado con funciones de matlab, filtran un resquicio en el primer impulso, permitiendo el correcto cálculo de la señal en energía (elección de tramo correcto).

4.3. CONCLUSIÓN Y ANÁLISIS FINAL.

Los resultados y los razonamientos anteriores, aportan una visión global del problema, que permite destacar las ventajas e inconvenientes de cada método, así como la conclusión de cuál debe ser el procedimiento a seguir para futuros estudios.

En la ilustración42 se muestra a grandes rasgos el análisis global del proyecto, para a continuación detallar cada método por separado.

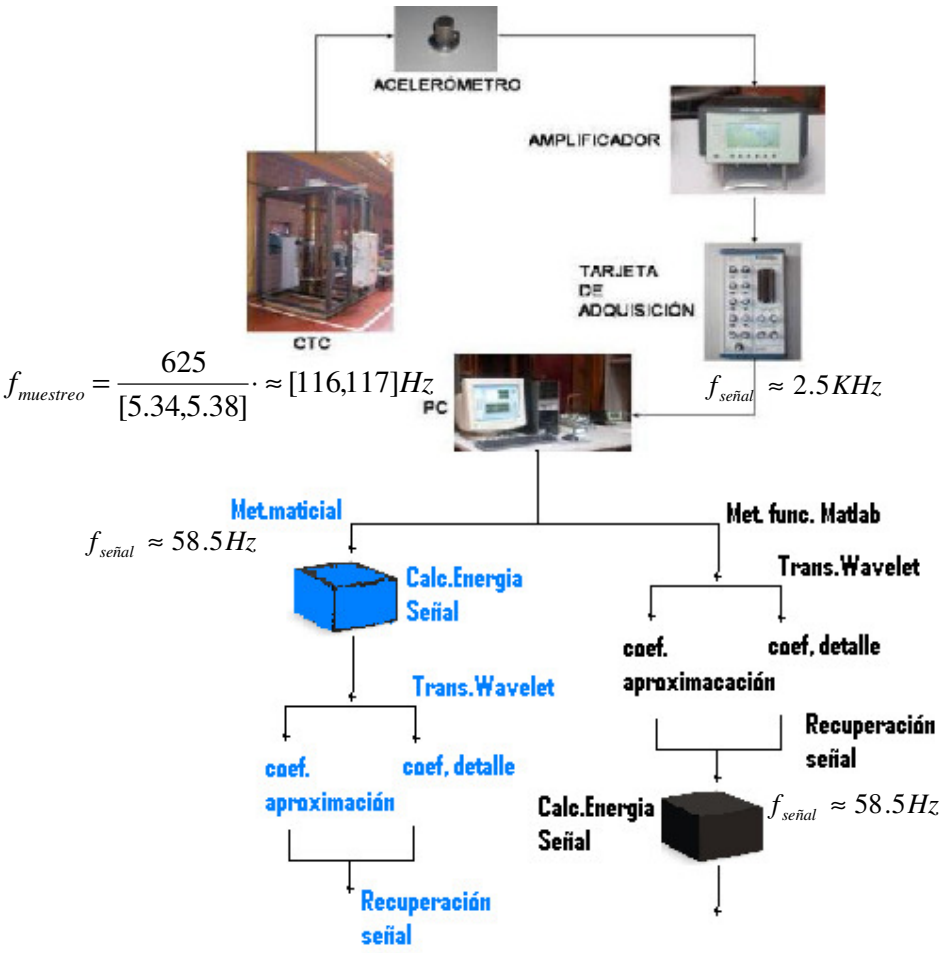


Ilustración 44. Resumen gráfico de ambos métodos

4.3.1. Análisis con métodos matriciales

Ventajas:

- ✓ Análisis transparente debido a las funciones implementadas.
- ✓ Se trabaja con señales optimizadas en cuanto número de muestras en energía. (Mínimo número de muestras para detectar 6 impulsos)

- ✓ Obtención de la señal con un número de coeficientes de wavelet bajo (640 coeficientes como máximo), además, se tiene la matriz capaz de recomponer la señal, luego sólo es necesario la realización de una simple operación para recuperar la señal de entrada.
- ✓ Método optimizado para la selección de coeficientes.
- ✓ Correcta detección de los impulsos en un alto porcentaje.
- ✓ Menor peso computacional que el análisis con funciones wavelet (debido al tamaño de la señal procesada).

Desventajas:

- ✓ No se puede trabajar con señales con gran número de muestras (debido a las matrices que se componen con la misma dimensión)
- ✓ No se puede realizar un análisis frecuencial detallado
- ✓ En algunos casos la detección es mas imprecisa que en el caso de hacer uso de las funciones de matlab.

4.3.2. Análisis con métodos implementados por matlab.

Es evidente que las ventajas y desventajas del método anterior se verán contrapuestas en este apartado, aún así se reseñan para una más completa explicación.

Ventajas:

- ✓ Posibilidad de hacer un análisis frecuencial exhaustivo: al trabajar con la señal en tiempo, se puede aplicar la transformada directamente, obteniendo un filtrado paso alto y paso bajo en cada nivel, y con los coeficientes, ser capaz de recuperar las frecuencias que interesen.
- ✓ Fiabilidad a la hora de obtener los impulsos correctos, ya que la señal filtrada, habiéndose eliminado impulsos espurios, permite una mejor obtención de la energía de los impulsos (eliminación de ruido).

Desventajas:

- ✓ Mayor coste computacional, al trabajar con señales con mayor número de muestras (5000000).
- ✓ Opacidad (caja negra) de las funciones de matlab que se usan.
- ✓ Los coeficientes que se obtienen son mucho mayores en número, siendo necesario, además, las funciones de matlab para la recuperación de la señal.

4.3.3. Conclusión final.

Es inevitable destacar la necesidad de ambos métodos para recapacitar acerca del correcto análisis y la correcta extracción de los parámetros de diagnostico.

Una vez detalladas las ventajas e inconvenientes, se plantea cual es la línea ideal a seguir, partiendo de la base de que se han alcanzado los objetivos planteados al inicio.

Para ambos métodos se tiene una correcta detección, aunque es cierto que en el caso de la señal 6_7_8_2.mat los métodos mostraban una ligera variación en el primer impulso.

Pues bien, toda vez, se tienen implementado ambos, se concluye que es necesario realizar el análisis por los dos métodos, para detectar si los datos obtenidos, son los idóneos para la clasificación de las posibles averías que se den en el transformador, y en el caso, de que no se tengan exactamente los mismos resultados con ambos métodos, usar los datos extraídos con el segundo de ellos (uso funciones wavelet de matlab).

Al final del documento, tras la explicación de cómo se ha realizado la interfaz de usuario, se muestra un apartado donde se aconsejan nuevas ampliaciones a este estudio y se detalla una vía paralela para llegar a los resultados anteriores.

CAPITULO 5. DESARROLLO DE LA INTERFAZ DE USUARIO.

5.1. OBJETIVO DE LA INTERFAZ

(Ref. [4]) Con el desarrollo de la interfaz, se pretende que cualquier usuario pueda realizar el análisis relatado anteriormente de una manera cómoda y sencilla, de tal forma que se seleccionen unos parámetros a la entrada, y se muestren los resultados alcanzados de manera que sean claros de entender y de fácil localización para poder usarlos en futuros estudios, así como en la etapa de clasificación de las averías.

Los parámetros de entrada que seleccionará obligatoriamente el usuario serán:

- ✓ Señal de entrada
- ✓ Tipo de análisis
  - Matricial
  - Funciones matlab
- ✓ Tipo de transformada de wavelet
- ✓ Máximo nivel a analizar

Los parámetros que se muestran son (parámetros de **salida**):

- ✓ Señal de entrada depurada en tiempo conjunto la señal depurada en energía
- ✓ Señales recuperadas de cada uno de los niveles
- ✓ Correlación entre la señal recuperada de cada nivel y la señal de entrada en energía.
- ✓ Impulsos detectados de la señal de entrada y de cada uno de los niveles.

5.2. JUSTIFICACIÓN

El motivo principal por el que se realiza la interfaz es evitar a cualquiera que desee realizar cualquier tipo de análisis, deba saber nada acerca de matlab, e incluso nada de acerca de cómo está realizado dicho análisis.

Es decir, se tratara de dotar al programa de un uso intuitivo del mismo, así como aligerar la manera de obtener los datos necesarios para el uso de los parámetros de salida, la redacción de características de estos datos o la rápida comprensión global del proceso.

Se hace pues obligatoria la redacción de un manual de ayuda, el cuál facilite el uso, evitando cualquier tipo de problema que pueda surgir en el transcurso del análisis.

5.3. ELECCIÓN DEL DISEÑO

Compone una parte muy importante de la interfaz, ya que dependiendo del diseño escogido se dotara de mayor o menor claridad y manejabilidad.

En un principio se optó por dividir la interfaz en dos partes, una primera en la que se seleccionaran los datos de entrada, y otra en la que se mostraran y se trataran los datos de salida.

En cualquier caso se ha tenido siempre claro los parámetros que deberían ser facilitados por el usuario así como los parámetros que deberían mostrarse.

En la ilustración43 se muestra el primer diseño realizado y en la ilustración42 el resultado su ejecución, donde se encuentran los parámetros de entrada descritos en el apartado 5.1.

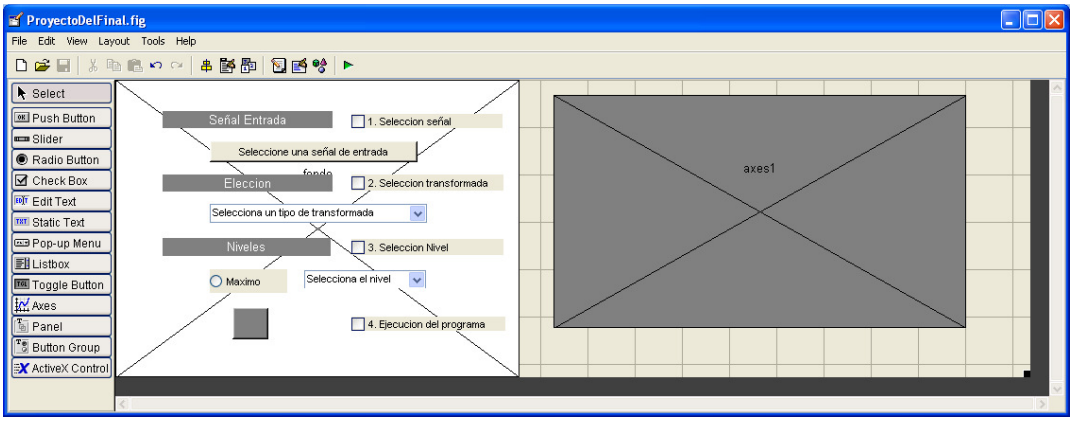


Ilustración 45. Diseño de la primera interfaz de entrada

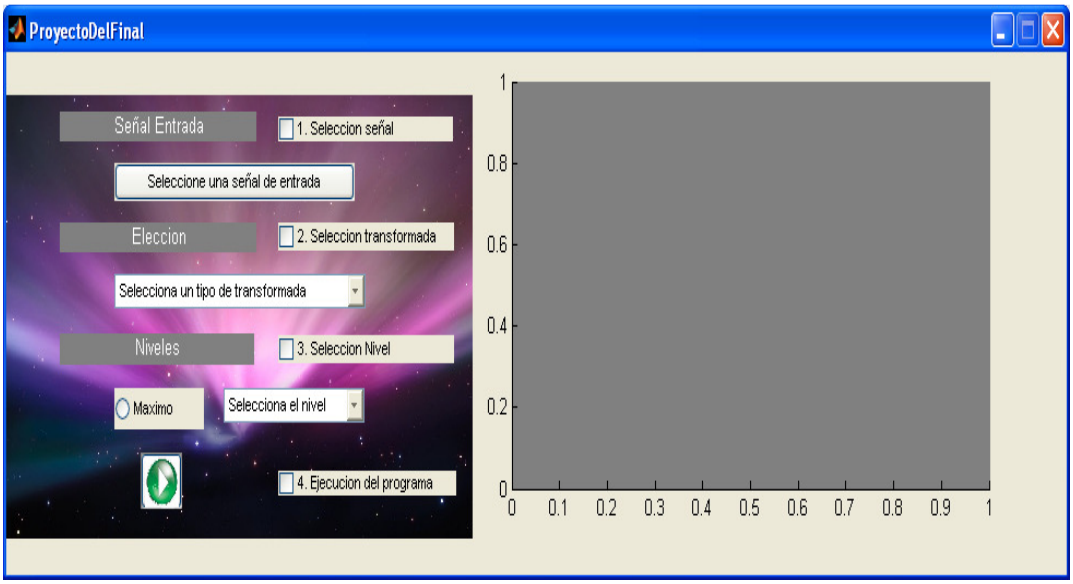


Ilustración 46. Ejecución del primer diseño de parámetros de entrada

Como se ve en la ilustraciónanterior, para que se realice el análisis de la señal, se debía de pasar por cuatro etapas, selección de la señal de entrada, selección de la transformada, selección de nivel, y pulsar el botón ejecutar.

Tras el análisis se ejecutaba otra interfaz cuyo diseño se presenta en la ilustración45.

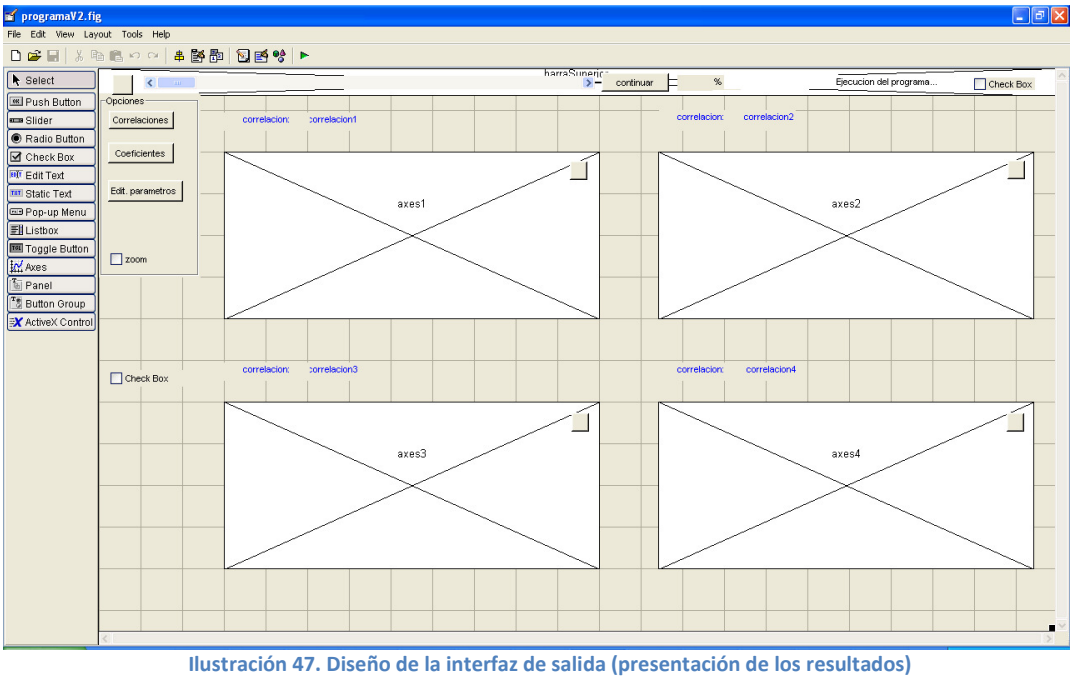


Ilustración 47. Diseño de la interfaz de salida (presentación de los resultados)

Otro diseño de la interfaz de entrada que se realizó pero se decidió no implementar es el mostrado en las figuras 46 y 47.

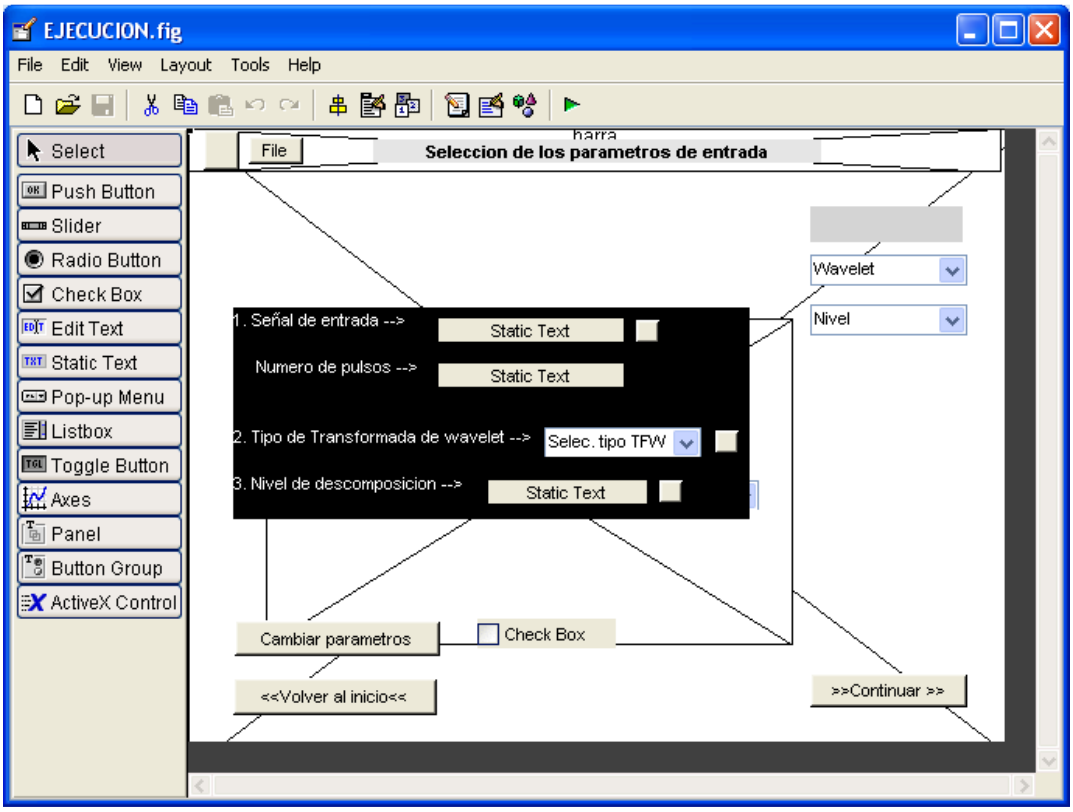


Ilustración 48. Diseño previo de la interfaz de entrada

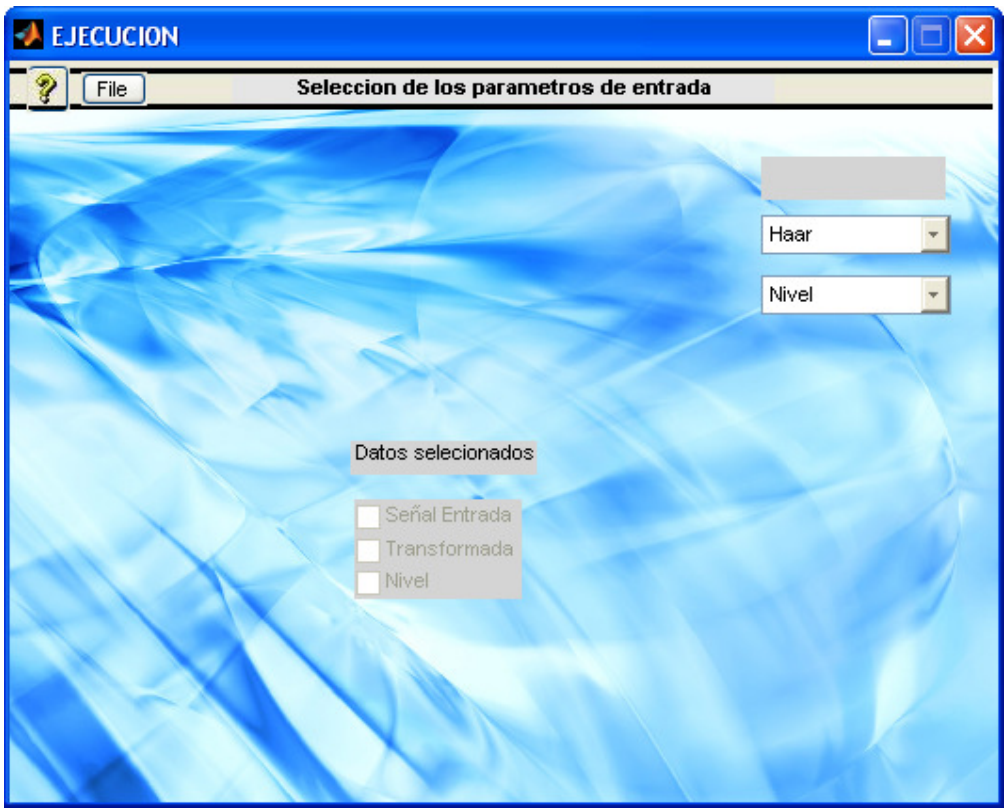


Ilustración 49. Ejecución del diseño de la interfaz de entrada

Los diseños mostrados, tenían en principio una fácil manejabilidad, y eran bastante intuitivos.

Sin embargo la posibilidad de adaptar el diseño al ya existente en matlab[5] hacía que se aprovecharan las cualidades de dicha herramienta, además de tener un guión claro de los objetivos a alcanzar, dentándolo de mayor facilidad a la hora de saber lo que se quiere y por tanto saber lo que se desarrolla.

Otra gran ventaja era hacer que toda la gente que alguna vez a usado wavemenu, pudiera tener esta otra herramienta, para el análisis de señales con cualidades que hacen que no se permitan analizar por medio de la herramienta de matlab.

Se agregó por tanto la funcionalidad pertinente, ya existente en el diseño anterior, y posteriormente se fueron añadiendo otras opciones que actualmente dotan a la interfaz de mayor calidad, como ejemplo de estas funciones tenemos:

- ✓ Barra de progreso
- ✓ Generación de reporte
- ✓ Barra con menús
- ✓ Ayudas
- ✓ Elección de análisis alternativos
- ✓ Pestaña indicadora de la evolución

En la ilustración48 se muestra el interfaz de la herramienta wavemenu, y en la 49 la interfaz desarrollada. Se pueden ver las grandes similitudes en cuanto al diseño, donde se ha decidido eliminar y dotar de las características deseadas, mencionadas anteriormente.

[5] wavemenu

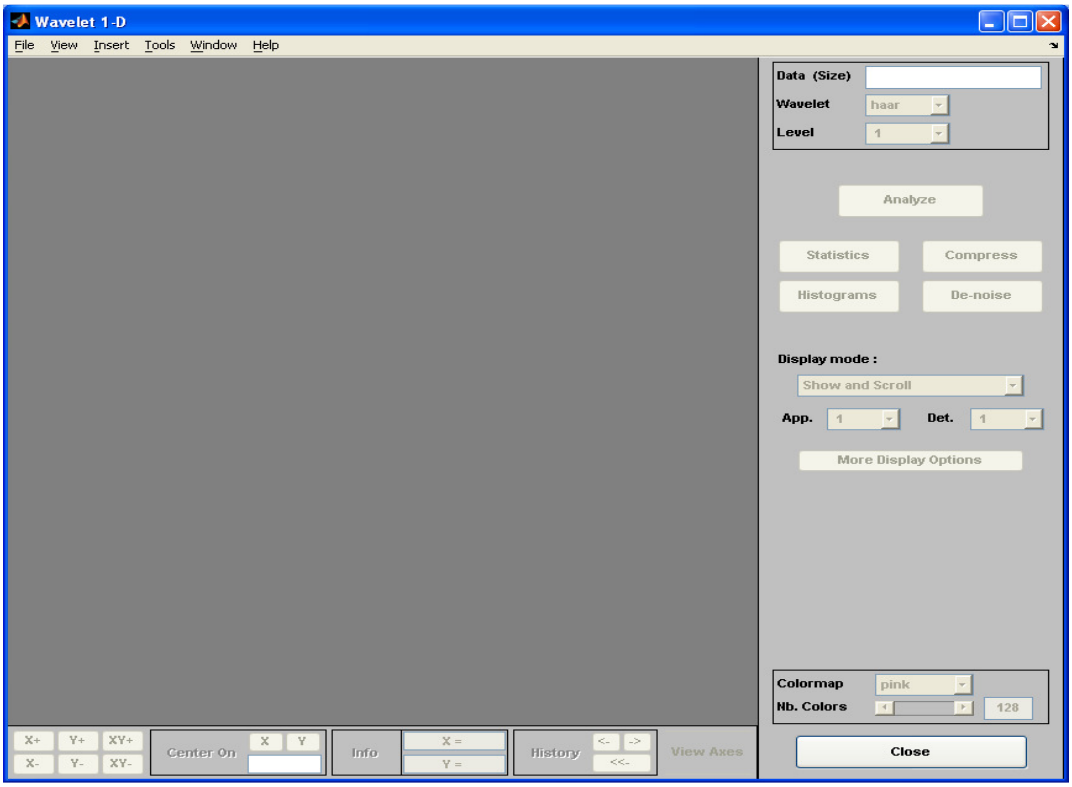


Ilustración 50. Interfaz de la herramienta wavemenu

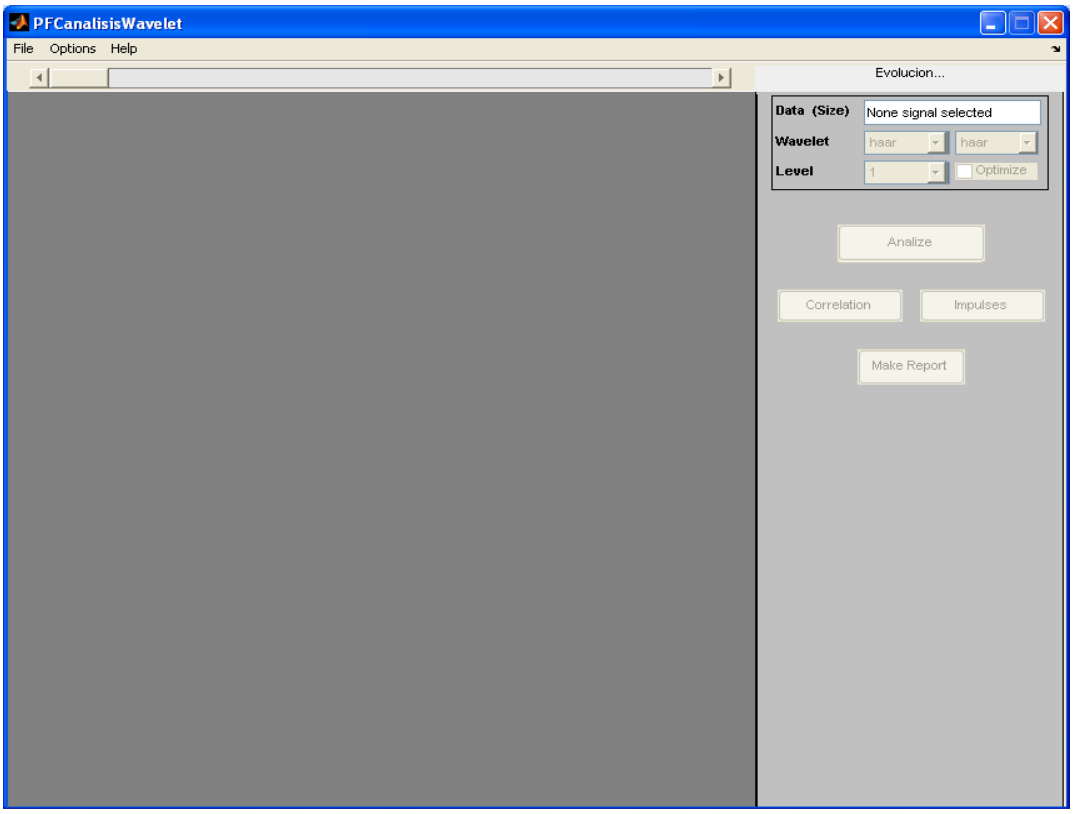


Ilustración 51 Interfaz final al inicio de su ejecución1

Se muestran habilitadas y a la vez deshabilitadas algunas pestañas y botones, como se explicará posteriormente cada pestaña o botón se habilita dependiendo de la fase del análisis en que nos encontremos.

Cabe destacar que, evidentemente, este diseño ha ido evolucionando poco a poco, por lo que en un inicio la interfaz lucia de otro modo.

En la ilustración50 se observa como se ha diseñado, dando lugar a la ilustraciónanterior.

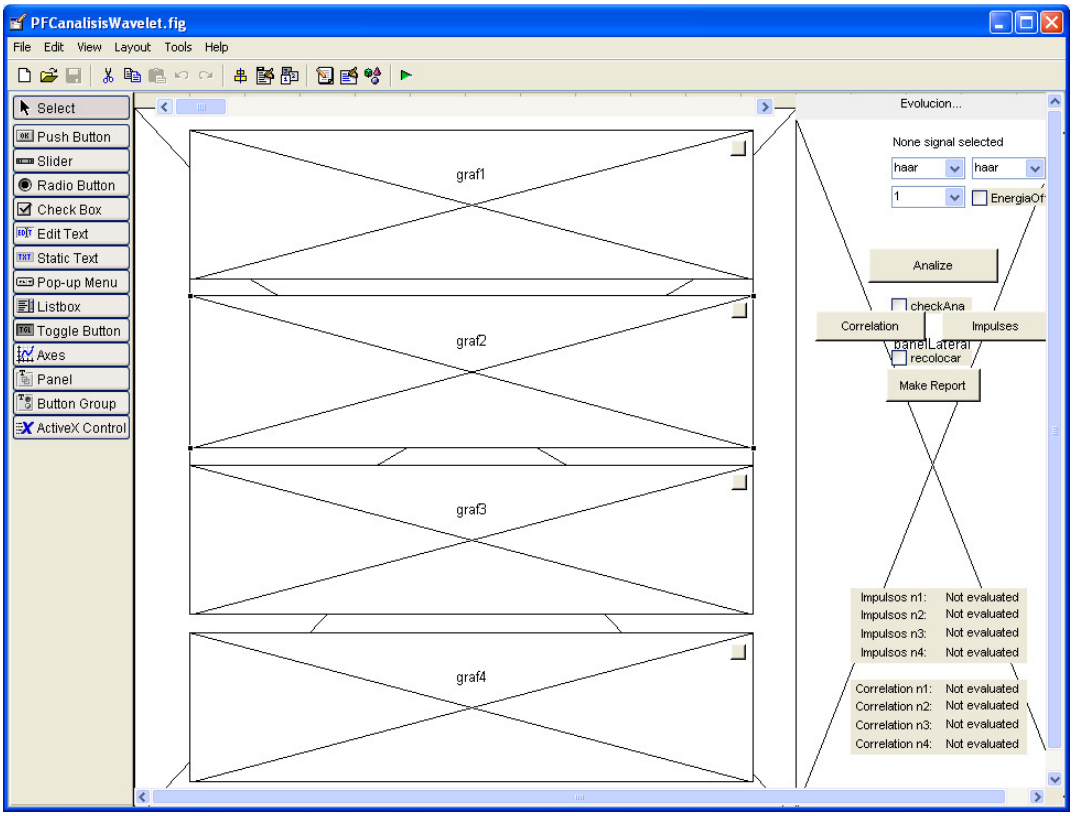


Ilustración 52 Diseño final de la interfaz

5.4. DESARROLLO DE LA INTERFAZ

Antes de detallar cada una de las partes de la interfaz, se hace necesario destacar, que cada una de las pestañas o botones habilitados, dispondrá de una función a la que se llamara cuando se actúe sobre dicho botón o pestaña. De esta forma se tiene que tener especial cuidado con todas las funciones que se ven afectadas entre sí.

Tras el desarrollo de la interfaz, se ha implementado un método de seguimiento, para identificar y solventar los problemas que han ido surgiendo, dichos bugs han sido clasificados, de forma que se destacarán los más significativos según se describa cada análisis.

Este apartado se divide a su vez en dos, parámetros de entrada y de salida, y dentro de cada parámetro se tiene:

- ✓ Cómo introducir este parámetro
- ✓ Operaciones implicadas
- ✓ Rango de valores permitidos
- ✓ Parámetros de salida
- ✓ Seguimiento de la interfaz, pasos siguientes.

Por último destacar la gran importancia que tiene, el correcto tratamiento de cualquier error que pueda ser provocado, presentándolo de una manera amable para el usuario, el cual se percate del error, y pueda introducir parámetros que se adecuen al diseño.

5.4.1. Acciones entrada salida: Selección de la señal.

- ✓ Introducción del parámetro

En la ilustración51 se muestran las acciones para introducir la señal: File→Load→Signal.

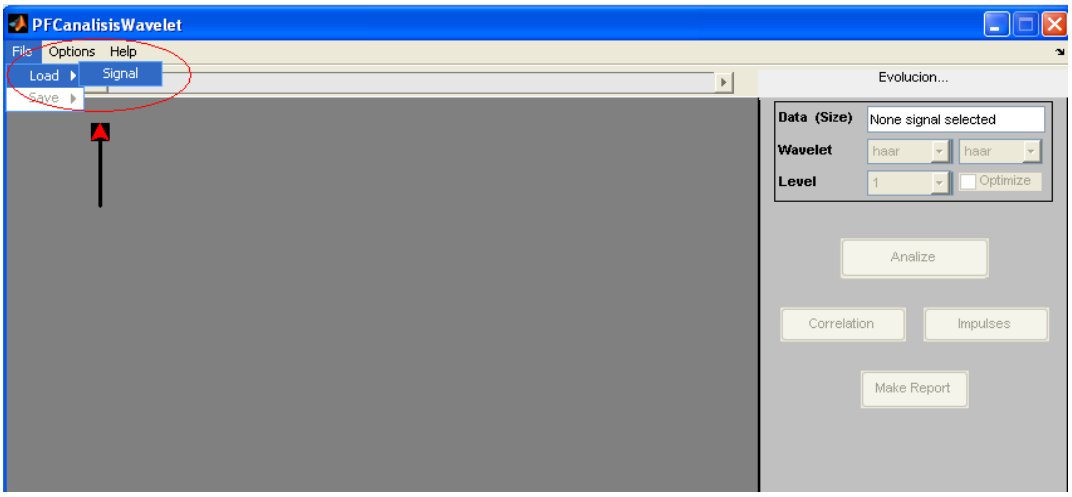


Ilustración 53. Selección de la señal de entrada

- ✓ Operaciones implicadas.

En principio se ha optado por el uso de una barra de menús, ya que la gran mayoría de aplicaciones que existen en el mercado, optan por este modo de ingresar los datos de entrada al inicio del uso del programa.

Ya que la interfaz, debe interaccionar con un programa base ya existente, el cual debe mostrar sus análisis en pantalla, se debe poner especial cuidado en el paso de parámetros a la interfaz. De este modo, existen ciertas variantes a la hora del paso de parámetros desde la aplicación base a la interfaz y viceversa, que harán variar el comportamiento de esta última.

Este hecho hará que se deban comprobar los parámetros de entrada en cada una de las aplicaciones para diferenciar que tipo de análisis se requiere o los datos que se deben mostrar en cada momento.

En este apartado, se tiene una casuística que muestra muy bien lo que se acaba de comentar.

La introducción de la señal de entrada provoca que en principio sólo tengamos un parámetro a analizar, por tanto, se debe indicar al programa base, que sólo se debe de hacer un primer análisis de la señal de entrada, que nada tendrá que ver con el análisis multirresolución de la trasformada, sino que simplemente se quiere hacer una primera detección de los impulsos y una depuración de la señal de entrada.

A este efecto, en primer lugar se comprueba, si se ha pulsado el botón de analizar, en caso de que no haya sido así, es porque todavía no se requiere un análisis global, por lo que se prepara la interfaz para mostrar la señal de entrada.

Cuando se introduce la señal, se activa la función que acompaña al botón correspondiente a “load signal”.

Esta función comprueba si es la primera vez que se carga la señal, o sin embargo ha habido un análisis previo, en este caso se debe eliminar la presentación que existe en pantalla, y prepararla para la siguiente. Este paso tiene vital importancia, ya que como se tiene en el apartado “tratamiento de gráficas” se debe poner especial cuidado es su posicionamiento, según el análisis que se vaya a realizar.

Una vez se ha preparado el entorno, sólo queda comprobar que la señal introducida es válida, y en caso de que no sea así, mostrar los mensajes pertinentes para que el usuario se percate de este error.

Este breve apartado se basará principalmente, en mostrar cómo se ha realizado esta comprobación.

Finalmente, se pasará a analizar la señal que se ha introducido. Con lo cual se debe llamar a la función base para que se realice el análisis.

Aquí es donde se comentaba, que se debe diferenciar este análisis previo del análisis global, para lo cual se llamaran a las funciones pertinentes, en este caso “depuración”.

✓ Rango de valores permitidos.

Ya que la señal que se mide directamente del intercambiador de tomas en carga, obtenido mediante el "data adquisition" posee varias columnas de las cuales se toma la quinta, una posibilidad que ofrece la interfaz, es introducir la señal sin ser tratada previamente.

La segunda posibilidad consiste en introducir la señal como un vector fila o columna, es decir, previamente se ha tomado la columna cinco y se a transformada en vector fila o columna.

Cualquier otro tipo de señal dará lugar a un error, el cual es tratado y mostrado de forma que el usuario se percate de que ha introducido una señal no válida, como se muestra en la ilustración52.

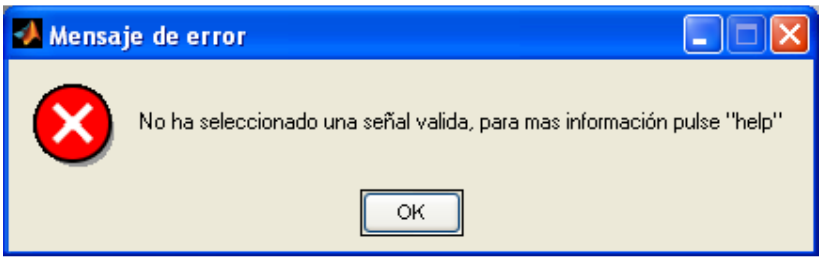


Ilustración 54. Mensaje de error, señal de entrada no válida

De igual forma, en el caso de que se decida cancelar la selección de la señal de entrada, se muestra el mensaje de la ilustración53, para advertir que no se ha seleccionado ninguna señal.

Ilustración 55. Mensaje de error

Por último, puede ser que se introduzca unos parámetros de entrada con dimensiones adecuados, pero que sin embargo no se corresponda con una señal válida, tal y como se espera en el desarrollo posterior, es por esto que en el transcurso del análisis, se intenta comprobar que la señal se ajusta a lo esperado, mostrando un mensaje de advertencia en caso de que no sea así, para que el usuario esté al tanto de la aparición de posibles errores.

Como ejemplo se muestra en la ilustración54, un mensaje de error producido por la espera de una eliminación de offset que se produce en

cualquier análisis y sin embargo en esta ocasión no se produce (debido evidentemente, a la introducción de una señal no válida)

Ilustración 56. Señal de advertencia

✓ Parámetros de salida

Una vez cumplimentado este paso previo para el análisis, se da la posibilidad de ver representada la señal de entrada, como vemos en la ilustraciónnumero 55.

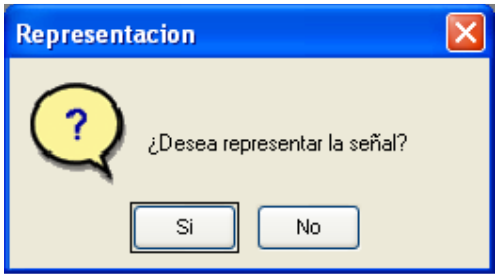


Ilustración 57. Cuestión en la representación

Respondiendo afirmativamente se muestra la señal representada, dándose la opción de ver los impulsos que se detectan en la señal de entrada, pudiéndose de esta manera comprobar, que efectivamente, la señal de entrada posee los seis impulsos que se pretender detectar tras el procesado.

Como se muestra en la ilustración53, una vez se tiene la señal de entrada, se muestra el nombre y el tamaño de esta, y se permite comenzar a introducir los parámetros propios del análisis. Como son, el tipo de transformada y el nivel.

✓ Seguimiento de la interfaz, siguientes pasos.

También se tiene la pestaña de optimización y el botón para analizar dicha señal. Por defecto tenemos el nivel uno y la transformada de wavelet tipo haar, ya que fue la seleccionada como wavelet madre en el diseño inicial.

Otro detalles a reseñar y que son implementados son: el avance de la barra de progreso y la descripción del paso en que nos encontramos,

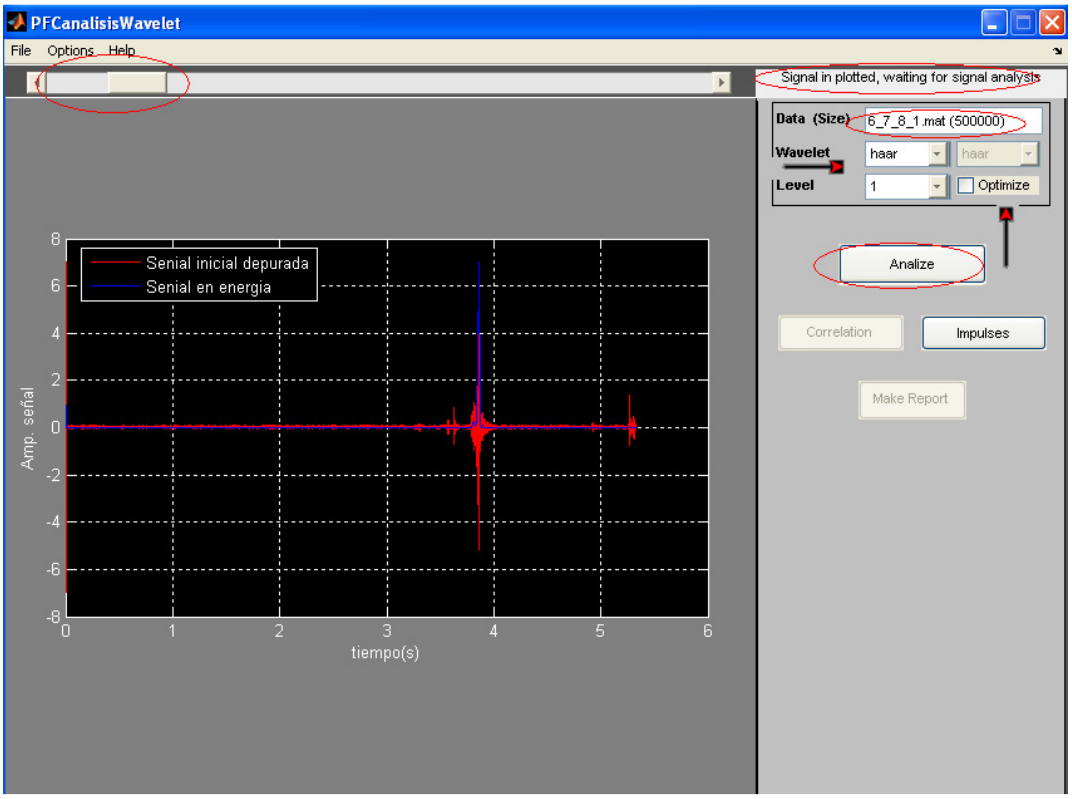


Ilustración 58. Resultados y progreso de la elección de la señal de entrada

5.4.2. Acciones entrada salida: Selección de nivel y tipo de transformada.

✓ Introducción del parámetro

Como se observa en la ilustración56 tenemos dos ventanas habilitadas para la selección del tipo de transformada y el nivel, y otra deshabilitada.

La primera pestaña nos permite la selección, en primer lugar del método matricial (que es el habilitado por defecto, del que se muestran el tipo de transformada) y en la última posición se tiene la posibilidad de la selección de habilitar el análisis con las funciones propias de matlab. Se muestran dichas opciones en la ilustración57.

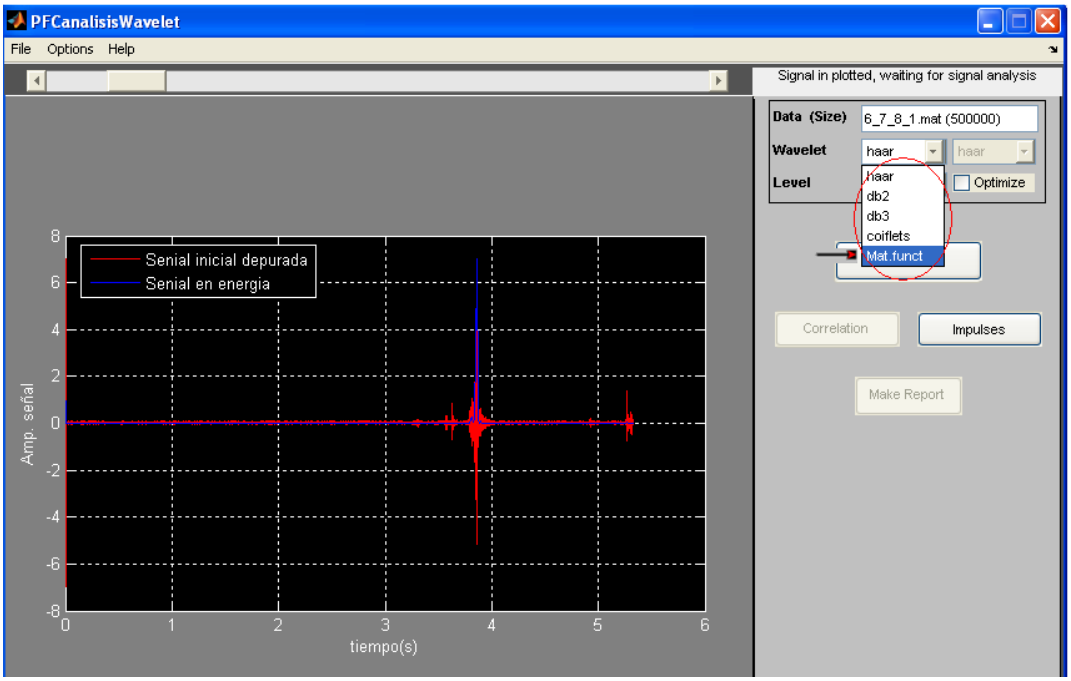


Ilustración 59. Tipo de análisis

Del mismo modo se muestra en la ilustración58 la selección del nivel máximo que se quiere analizar, que el tipo de análisis matricial es 4.

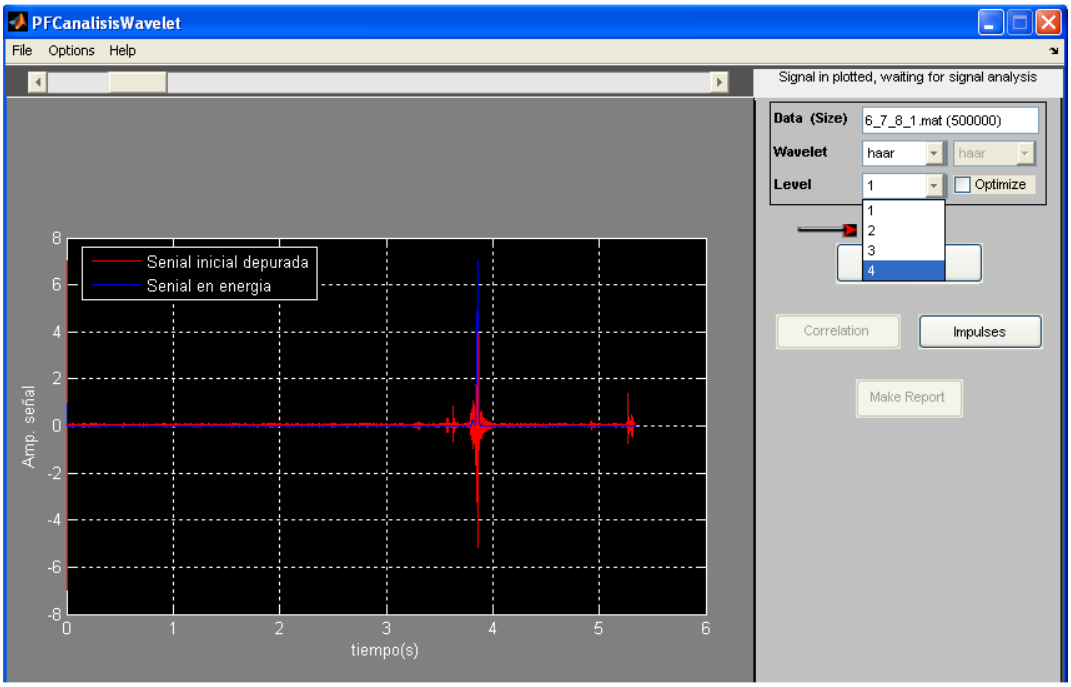


Ilustración 60. Selección de nivel

Cómo última alternativa a la selección de parámetros del tipo matricial, se tiene la posibilidad de marcar la pestaña optimize, mostrada en la ilustración56. Como se observa se deshabilita en la elección del nivel (ya que en este tipo de análisis siempre es el máximo.

De esta forma se hace un análisis hasta el nivel máximo, se muestran sólo los análisis que hayan dado lugar a un buen análisis, y se remarcan en la gráfica los impulsos, como veremos posteriormente.

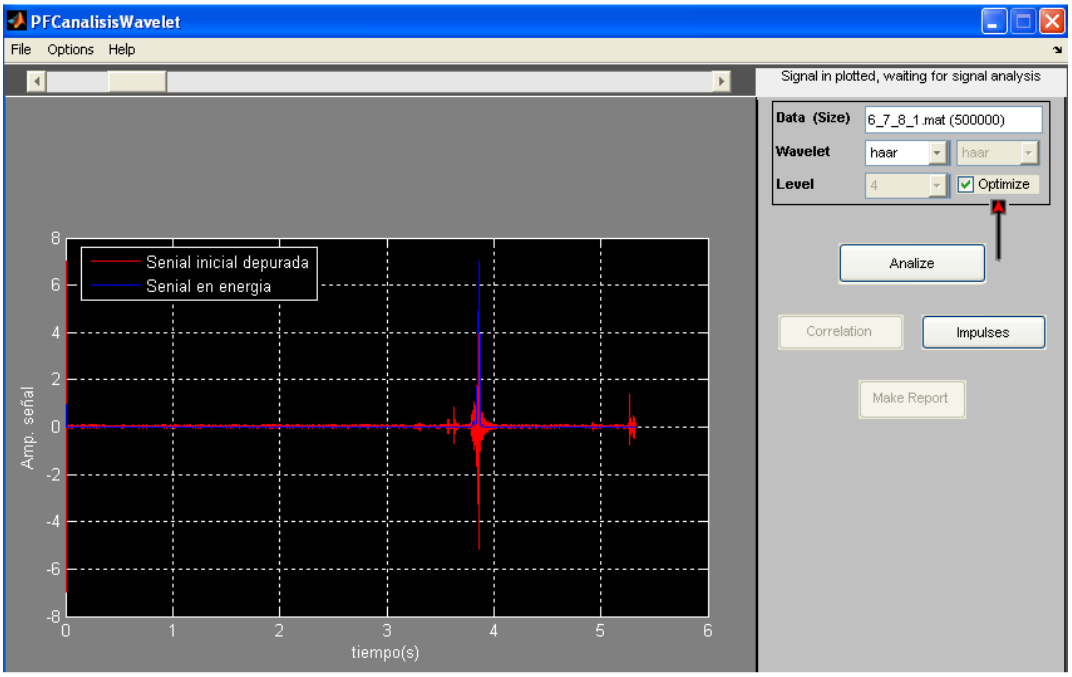


Ilustración 61. Opción de optimizar el análisis

Se pasa a continuación al análisis de los parámetros a seleccionar en caso de que se tome opte por analizar la señal de entrada con las funciones de matlab.

En la ilustración56 se muestra como tras la elección de hacer uso de las funciones mencionadas, se habilita la pestaña de la derecha para introducir el tipo de transformada, en este caso tenemos un tipo de transformada mas, el tipo biorthogonal.

Otro detalle que se muestra en la ilustración59 es el cambio de la pestaña optimize por la etiqueta “energiaOff”, esto es debido a que se tienen dos tipos de caminos posibles dentro del análisis haciendo uso de las funciones de matlab, trabajando con la señal en energía (lo cual se desaconsejo, ya que no se obtienen los resultados esperados) o con la señal en tiempo.

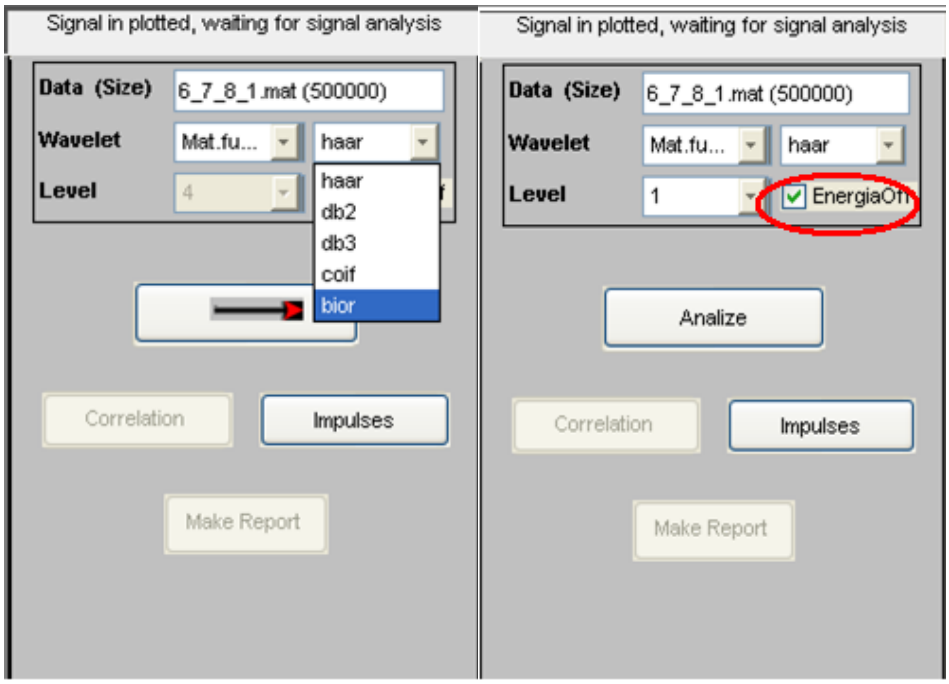


Ilustración 62. Cambios respecto al análisis matricial

En caso de seleccionar la pestaña de energiaOff, se habilitan un mayor número de niveles para realizar el análisis.

✓ Operaciones implicadas

Las opciones implicadas en este caso, tienen que ver con la activación y desactivación de pestañas, ya que no todos los tipos de análisis permiten la entrada de los mismos parámetros y en el mismo rango.

En cada una de las posibilidades de análisis se debe tener muy claro que parámetros son necesarios en cada caso, cuales se deben mostrar, habilitarse o cambiarse de valor.

Situación inicial: En este caso tendremos deshabilitada la pestaña de selección de transformada de la derecha, ya que se usa para el caso del uso de las funciones matlab. Además en ambos ejemplos tenemos el botón "correlation" y "make report" deshabilitado, hasta que se produzca el análisis.

La pestaña donde se muestran los niveles que se pueden elegir va desde 1 a 4, y la pestaña optimize está habilitada y con valor 0.

Opciones: Tenemos la opción de seleccionar el botón "optimize" provocando que los nivele se fijen a 4, y se deshabiliten, volviendo a habilitarse en caso deseleccionar la pestaña.

Otra posibilidad es seleccionar las funciones matlab, donde se muestran los cambios más significativos, son la habilitación de la selección de transformadas que aparece a la derecha de la que se usa para el análisis con operaciones matriciales.

Donde anteriormente teníamos la pestaña "optimize", ahora aparece la pestaña "energyOff", esta operación implica que se oculte la pestaña anterior y se pueda visualizar la actual. Esta pestaña hará que aparezcan mas niveles posibles de análisis.

En caso de que de nuevo se seleccione de nuevo una transformada del tipo de análisis matricial, se volverá a los parámetros iniciales.

Como se puede denotar, estas combinaciones hacen que en más de un caso se hayan encontrado errores a la hora de darse las diferentes combinaciones de habilitación y des habilitación, hacer visibles los elementos etc... (posteriormente se mostrará un ejemplo de ello).

✓ Rango del valores permitidos

En este caso, el usuario sólo puede seleccionar valores preestablecidos, con lo que no se hace necesario comprobar la validez de dichos parámetros.
Sin embargo, si se debe mostrar especial cuidado, con las posibles combinaciones que permitan habilitar y deshabilitar ventanas, se cita el caso de cambio a funciones matlab, lo que provoca que se habilite la pestaña de la derecha y se cambie la pestaña de optimize, operación que se ha de realizar a la inversa si se vuelve al análisis matricial.

Como bug localizado en este apartado, lo cual puede servir de ejemplo para cualquier que se aventure a realizar una interfaz de este tipo, se tuvo lo siguiente, (se muestra a continuación, del bug conforme se reportaba y se trataba en un documento excel):

Seleccionando el tipo de análisis: 1. Dentro del tipo matricial, seleccionar optimize. (La pestaña de selección de nivel se fija a 4 y se bloquea) 2. Seleccionar tipo de análisis Func. Matlab (se vuelve a habilitar la selección de niveles) 3. Se sigue mostrando bloqueada la pestaña de selección de nivel	Fijado	**Solución**: Si se selecciona en las funciones matlab, ocultar la selección de niveles para funciones matriciales, habilitándola y poniéndola de nuevo a 1, para el siguiente análisis y mostrar el nivel para funciones matlab, set(handles.nivelAux,'Visible','on') set(handles.nivel,'Enable','on') set(handles.nivel,'Visible','off') set(handles.nivel','Value','1')

✓ Parámetros de salida

Una vez se han seleccionado los parámetros necesarios para realizar el análisis se deben mostrar las gráficas correspondientes a las señales entrada y recuperada, con cada uno de los tipos.

Como los datos que se muestran son los mismos para los dos tipos de análisis, en la ilustración61 se muestran las salidas de uno de los análisis, en este caso se ha seleccionado el análisis matricial, 4 niveles, tipo Haar.

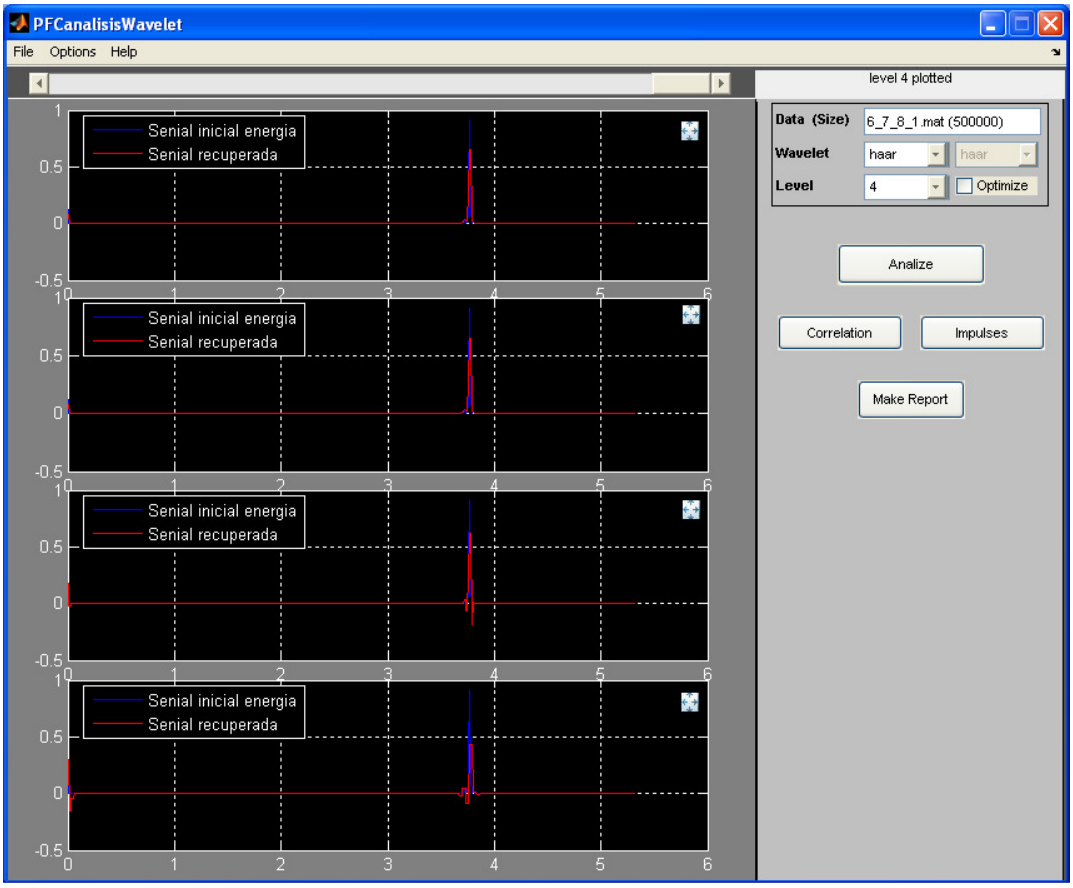


Ilustración 63. Datos de salida de análisis

Por otra parte si hay que resaltar como se comentó anteriormente una ligera variación en la salida en caso de seleccionarse la pestaña optimize como se observa en la ilustración62.

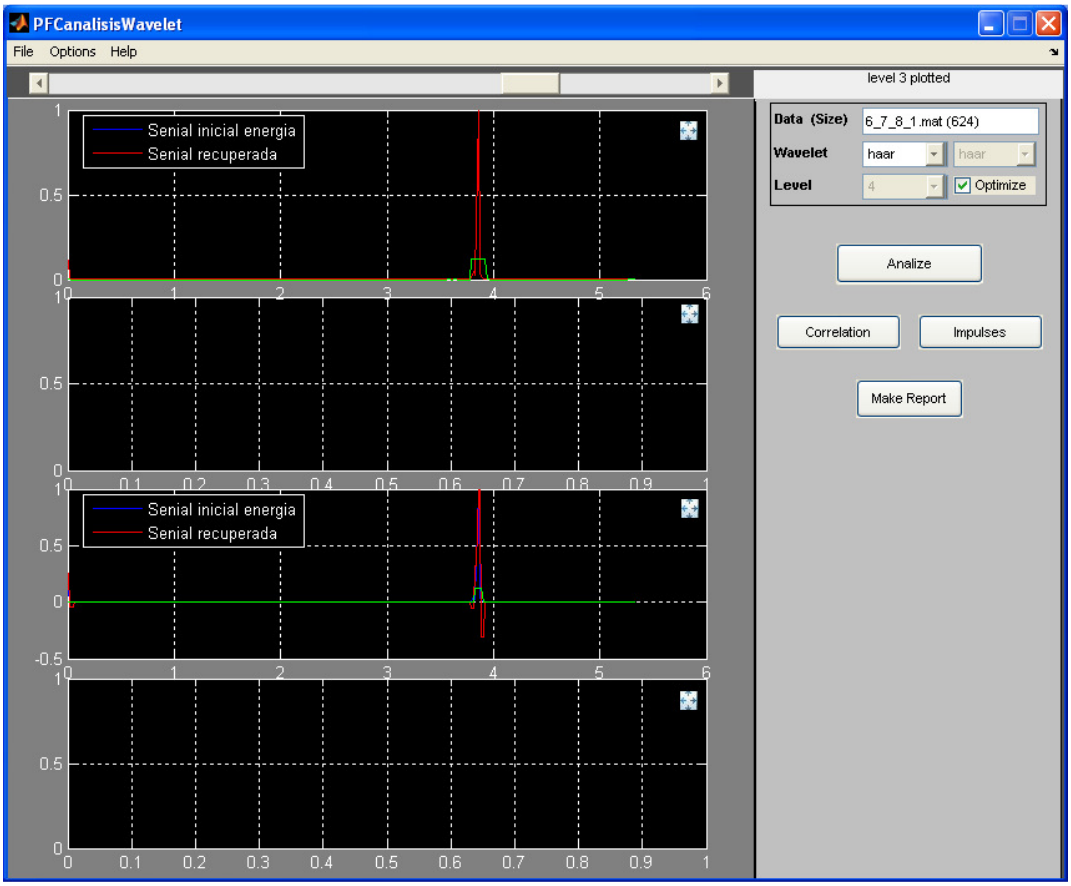


Ilustración 64. Resultados de salida seleccionando la pestaña optimize

✓ Seguimiento de la interfaz: Siguientes pasos

Los posibles pasos siguientes, son obtener la correlación y los impulsos de las señales analizadas. Evidentemente en el último caso analizado, al seleccionarse la pestaña optimize los impulsos serán 6.

Sin embargo para saberlos en el resto de los casos, basta con pulsar los botones correlación e impulses para saber la correlación entre la señal de entrada y la recuperada e impulsos respectivamente.

5.4.3. Acciones entrada salida: mostrar correlación y número de impulsos.

✓ Cómo introducir este parámetro

Como es lógico basta con pulsar los botones “impulses” o “correlation” situados por debajo del botón analize, mostrados en la ilustración63.

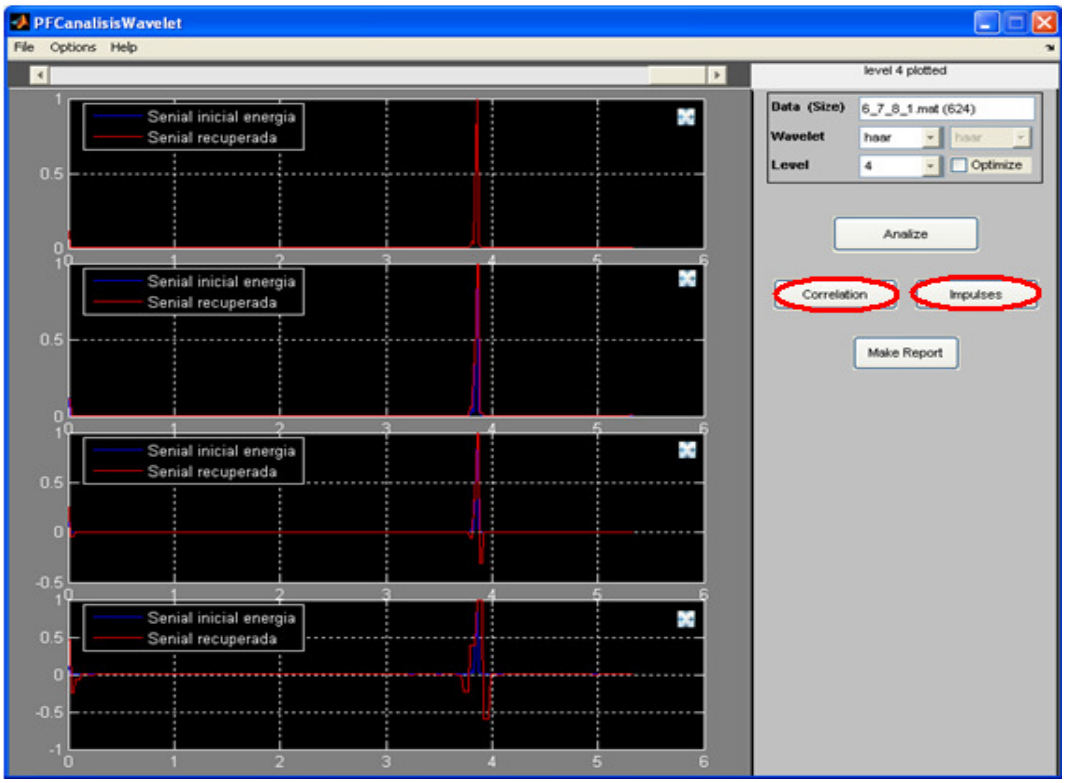


Ilustración 65. Botones correlation y analize

✓ Parámetros de salida

Una vez se pulsan los botones se muestran en pantalla las correlaciones y los impulsos de las señales, en caso de que no se hallan seleccionado todos los niveles, sólo se muestran los analizados.

De la misma forma cuando comienza un nuevo análisis o cuando se introduce una nueva señal de entrada se borran los valores anteriores, pero se dejan los títulos mostrados, no mostrándose de nuevo los resultados de la señal analizada, a menos que se pulsen de nuevo los botones. En las figuras 64 y 65 se muestran ambos análisis.

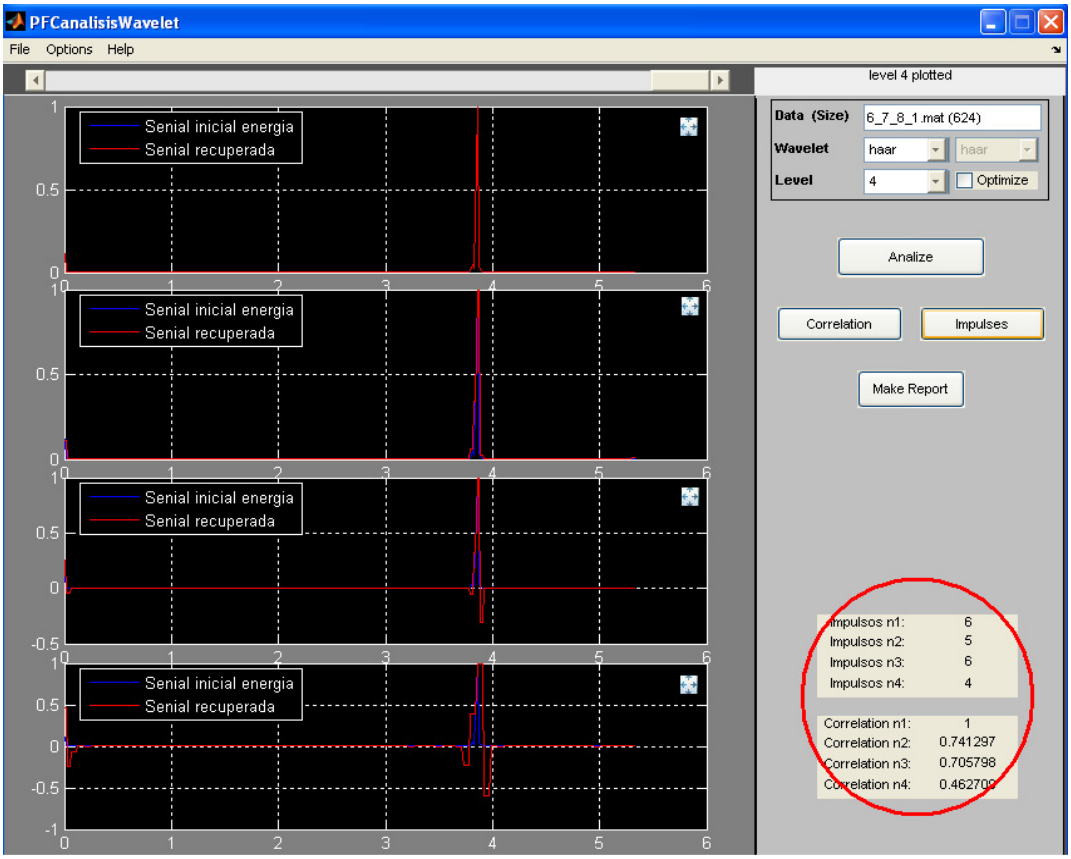


Ilustración 66. Correlaciones e impulsos

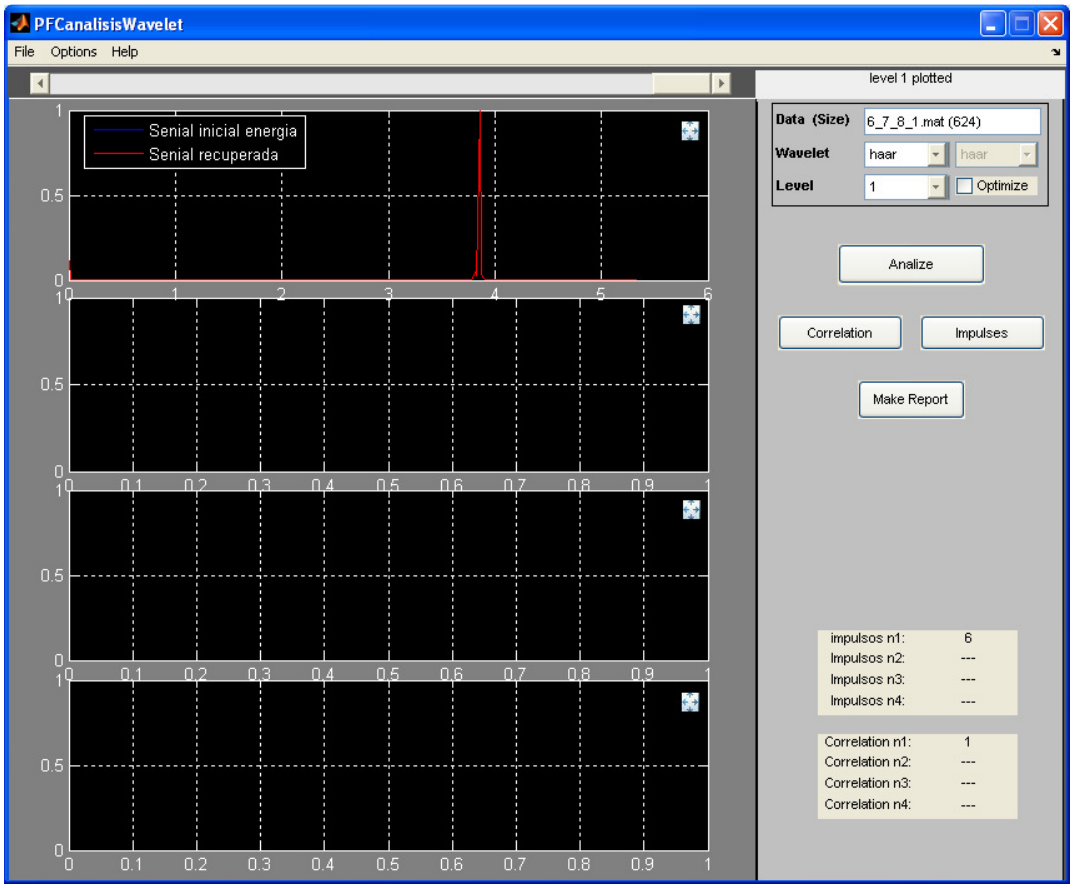


Ilustración 67. Borrado de los impulsos y correlación

✓ Seguimiento de la interfaz, pasos siguientes

Una vez tenemos las gráficas podemos seleccionar aquella que queramos ampliar, este será el próximo botón a destacar, conjunto las funciones que se realizan para el posicionado de las gráficas.

5.4.4. Acciones entrada salida: Operaciones de ejes y gráficas

Tal vez este sea uno de los apartados donde mayor cuidado se debe poner. Esto es debido a que los ejes no se pueden eliminar y además teniendo la opción de ampliar las graficas, se deben eliminar todas las que se muestran en pantalla, al igual que si se realiza un nuevo análisis.

Es por esto que se seleccionan posiciones fuera de la pantalla para las gráficas y según se necesiten se van colocando o descolocando de nuevo.

✓ Cómo introducir este parámetro (variaciones de colocación de las graficas)

En este caso tenemos varias acciones que hacen que se posicionen las graficas y ejes en distintas posiciones.

1. Representación de la señal de entrada→Eje grande
2. Representación de los niveles analizados →Señales, ejes pequeños
3. Aumento de eje pequeño→Eje grande
4. Cliqueo del botón de aumento de nuevo → Eje pequeño

En la ilustración66 mostramos la pestaña de aumento en sus dos posibles posiciones.

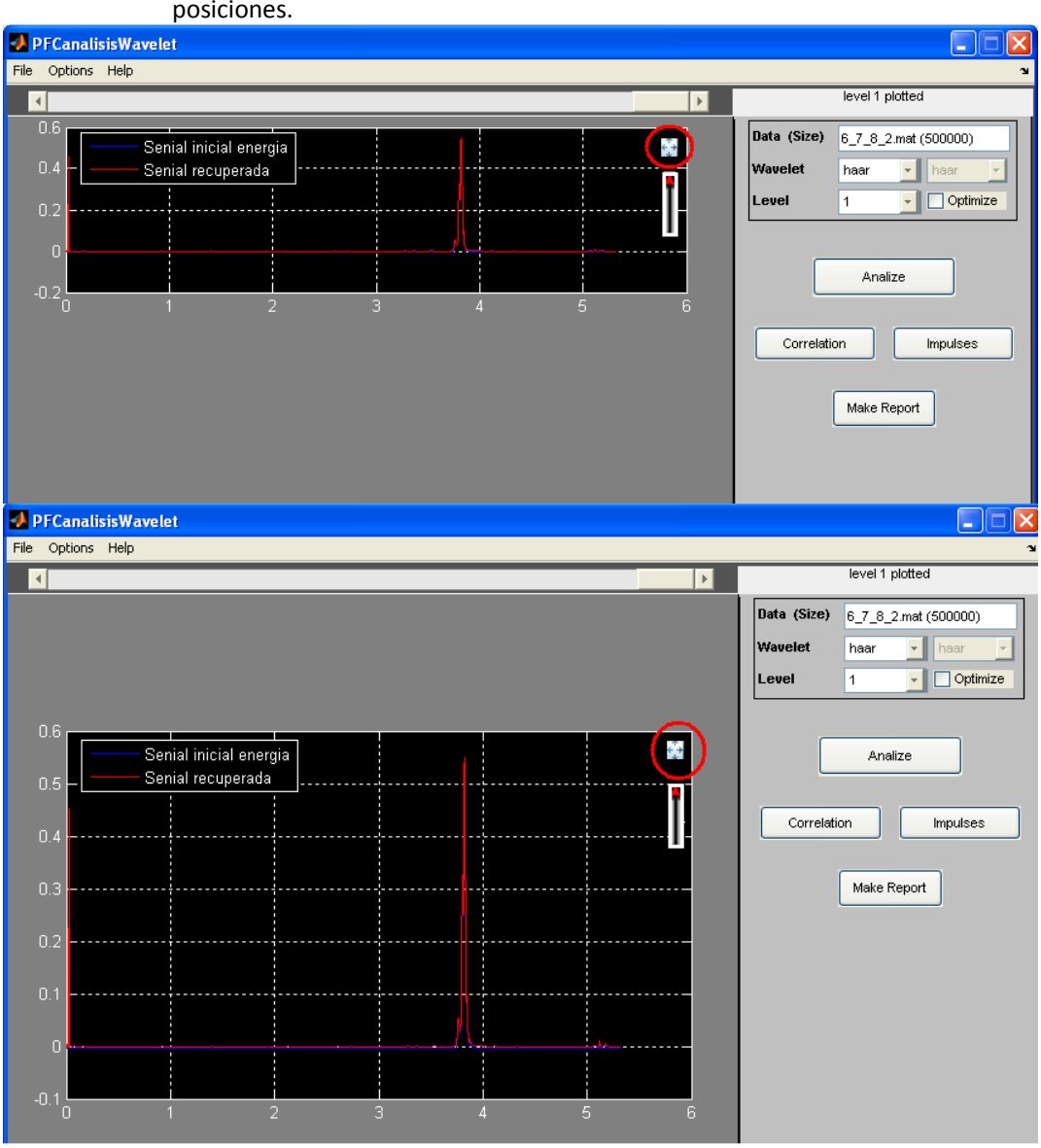


Ilustración 68 Posiciones del botón de aumento y cambio de graficas

✓ Parámetros de salida

Como se comentó en el apartado anterior tenemos varias posiciones de las gráficas. Como se ha visto, en la ilustración67, tenemos dos configuraciones, a continuación en la ilustración68 se mostraran los cuatro posibles botones de aumento y la desaparición de los mismos cuando se analiza una nueva señal de entrada.

En este apartado, como se comentó, se debe poner especial atención en el posicionamiento de las figuras, así como el hacer invisible los botones que nos sea pertinente mostrar.

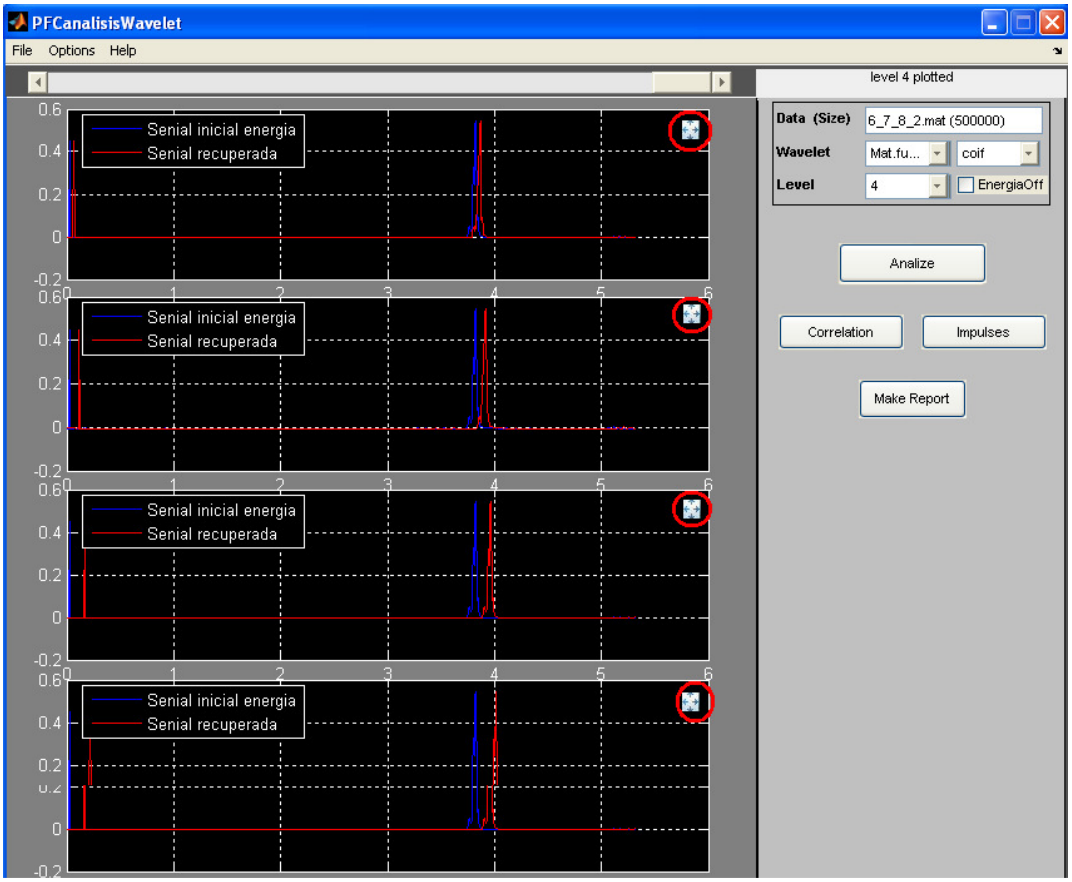


Ilustración 69 Muestra de los 4 botones de aumento que existen

✓ Seguimiento de la interfaz, pasos siguientes

Ya sólo queda por mostrar unas pocas características más, entre ellas la opción “make report” que será el paso siguiente en la descripción de la interfaz.

5.4.5. Acciones entrada salida: Herramienta make report.

Esta herramienta, se usa para generar un reporte con las características principales del análisis, y guardarlas en un archivo .doc para su posterior uso.

A continuación se pasa a describir cada una de las partes que componen la generación del reporte.

✓ Cómo introducir este parámetro

Para que genere el reporte, simplemente se debe pulsar el botón "make report", sin embargo, hay que tener en cuenta que sólo se tendrá la posibilidad de crear el reporte, toda vez el análisis haya sido realizado.

En la ilustración68 se muestra la localización del botón.

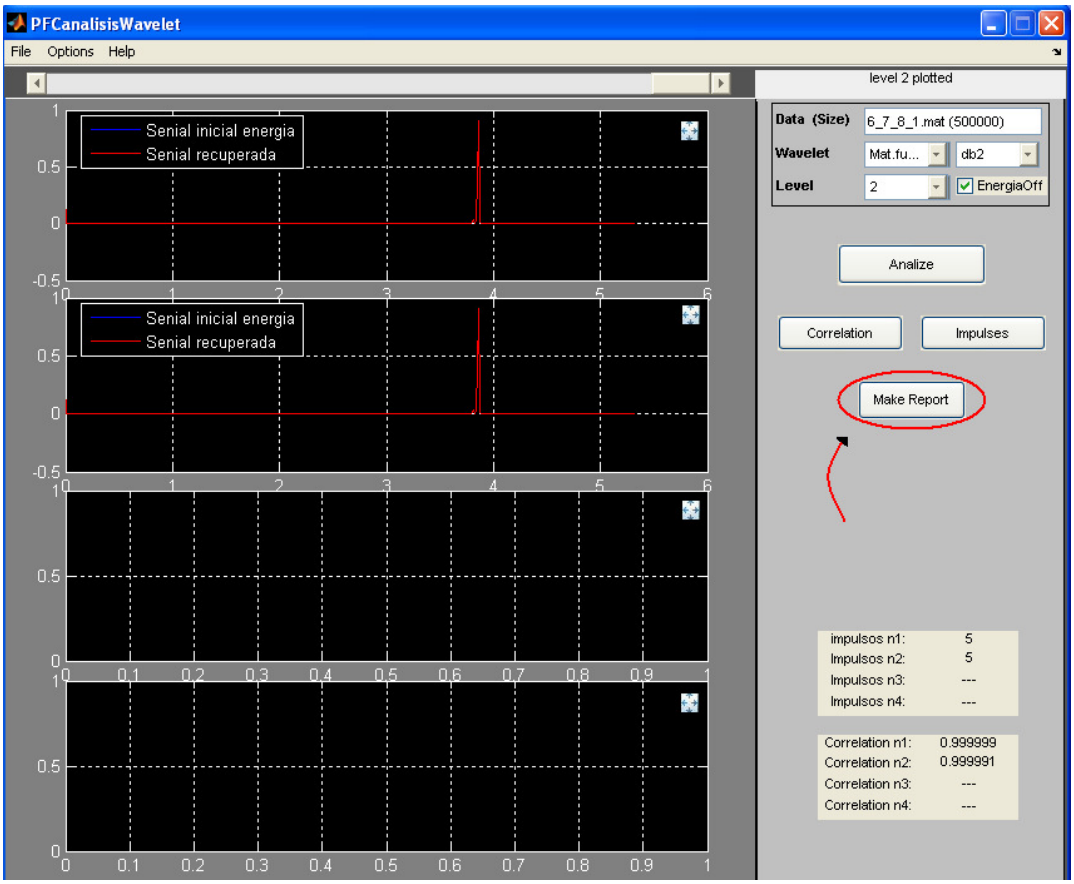


Ilustración 70. Localización del botón "Make Report"

✓ Operaciones implicadas (capitulo 1)

Este apartado, lleva implícito el uso de una nueva herramienta de matlab: "MATLAB report generator".

Esta herramienta, la cual es accesible por medio del botón "start" de matlab, permite generar un reporte haciendo uso de la interfaz mostrada en la ilustración69.

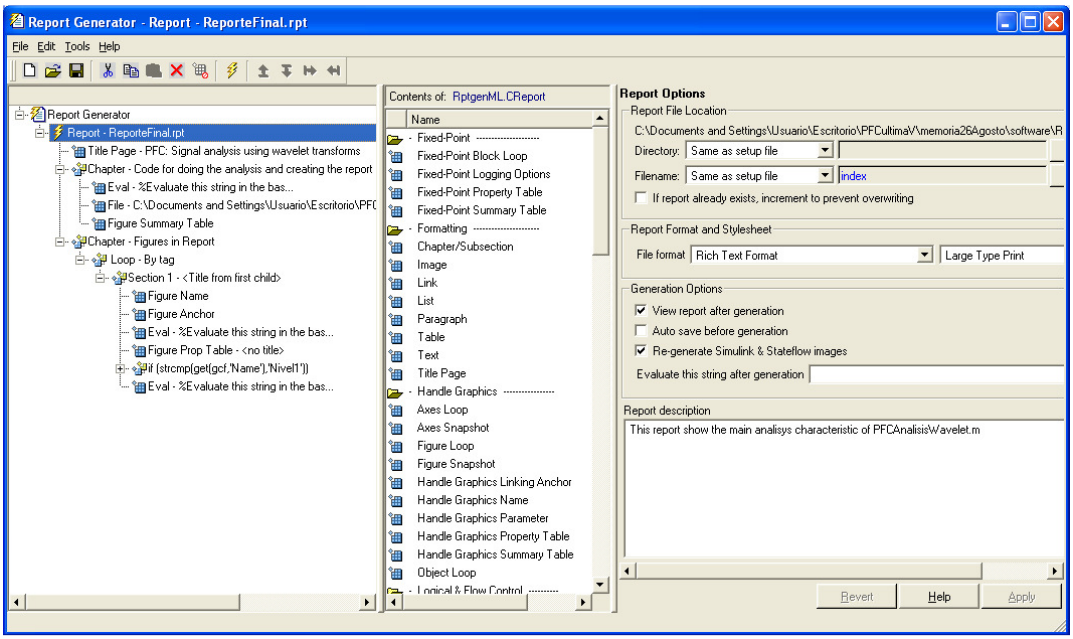


Ilustración 71. Interfaz report generator

En la ilustraciónanterior, se muestra de hecho, un ejemplo anterior al reporte final.

Supongamos hacemos una división de tres partes, dentro de la ilustración67.

A la izquierda, se tiene el flujo que ha de seguir la herramienta para generar el reporte, en el centro se tienen los elementos que se pueden agregar al flujo del programa, y a la derecha tenemos la características de los componentes a agregar.

Algo que se debe tener muy claro a la hora de la realización del reporte, es que se quiere obtener de él.

En este caso, se ha optado tras la realización y observación de varios reportes, por la impresión en un documento Word, de una tabla donde se muestra:

1. Tipo de transformada que se está llevando a cabo
2. Nivel de análisis
3. Impulsos detectados con el análisis
4. Correlación respecto a la señal de entrada.
5. Y en caso de que la detección haya sido correcta: energía y tiempo entre impulsos.

De la misma forma se puede hacer una portada, definición de capítulos, lo cual, pese haberse desarrollado en un primer momento, se decidió prescindir de estas características e ir directamente a la muestra de los resultados.

Se irá mostrando cada uno de los pasos que se han ido dando para su desarrollo, ya que en contraposición a la parte del desarrollo propio de la interfaz (cuya explicación metódica a través de las líneas de código que se han implementado, llevaría mucho tiempo y se escapa fuera de los márgenes del proyecto), esta parte es mas clarificadora, y supone un plus al escrito.

Una vez abierta la herramienta, *el primer paso* es indicar, que tipo de fichero es el que se va a generar. En este caso se ha seleccionado testo enriquecido y single print.

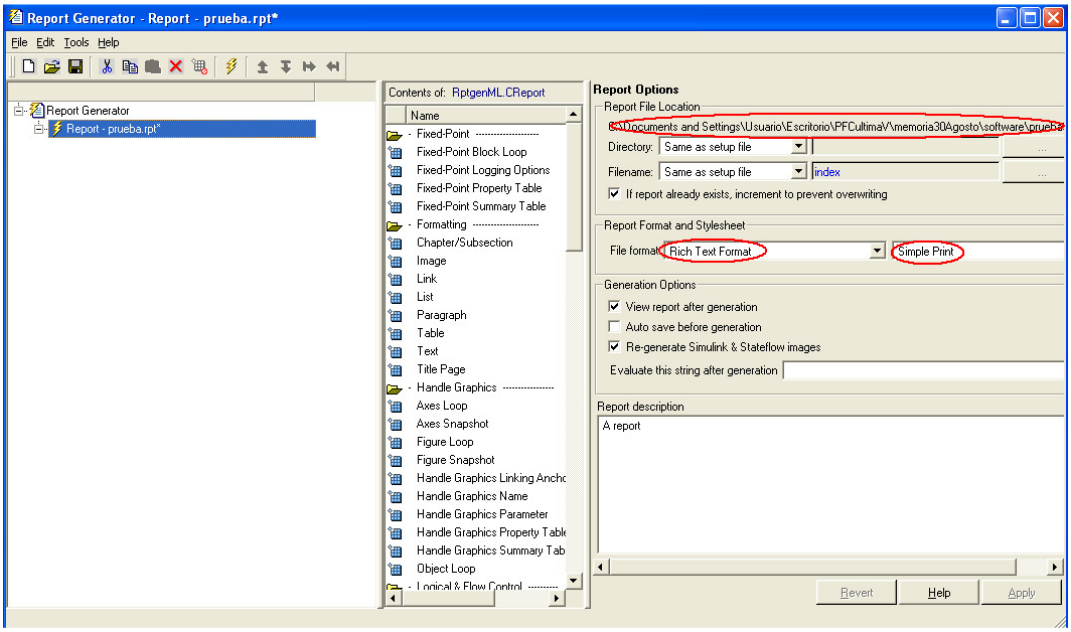


Ilustración 72. Creación del reporte

En la ilustración70 también se ha seleccionado el mismo directorio para guardar el reporte.

A la vez que se generará en .doc, también se tendrá un archivo html. Como se verá a continuación el uso de testo enriquecido traerá consigo alguna incompatibilidad.

En segundo lugar, se creará el primer de los dos capítulos, para la presentación de las cuatro primeras características numeradas en el apartado anterior.

Para la creación del capítulo usamos el objeto Chapter tal como se muestra en la ilustración71.

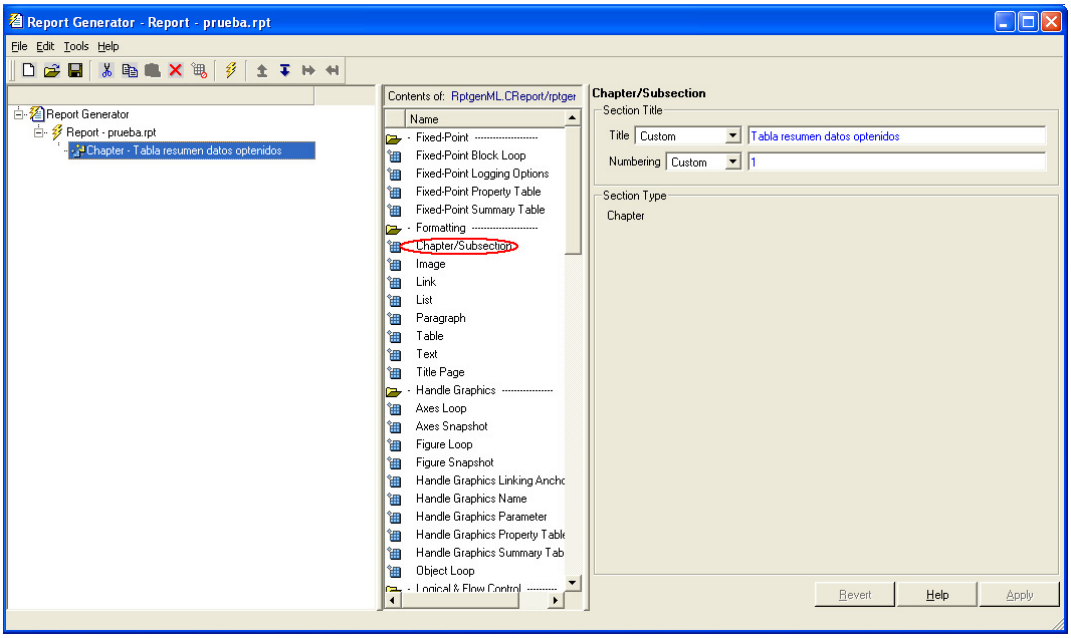


Ilustración 73. Selección del capítulo 1

En tercer lugar, como se describió anteriormente, se debe mostrar el tipo de método que estamos usando para la obtención de los coeficientes (matricial o a través de funciones matlab) y el tipo de transformada que se está usando.

Para ello como para el resto de las variables que se usan, previamente a la realización del reporte, se han guardado en un fichero .mat, de esta forma, cuando creamos la variable que se debe de adjuntar en el reporte, se seleccionan los parámetros que se muestran en la ilustración72.

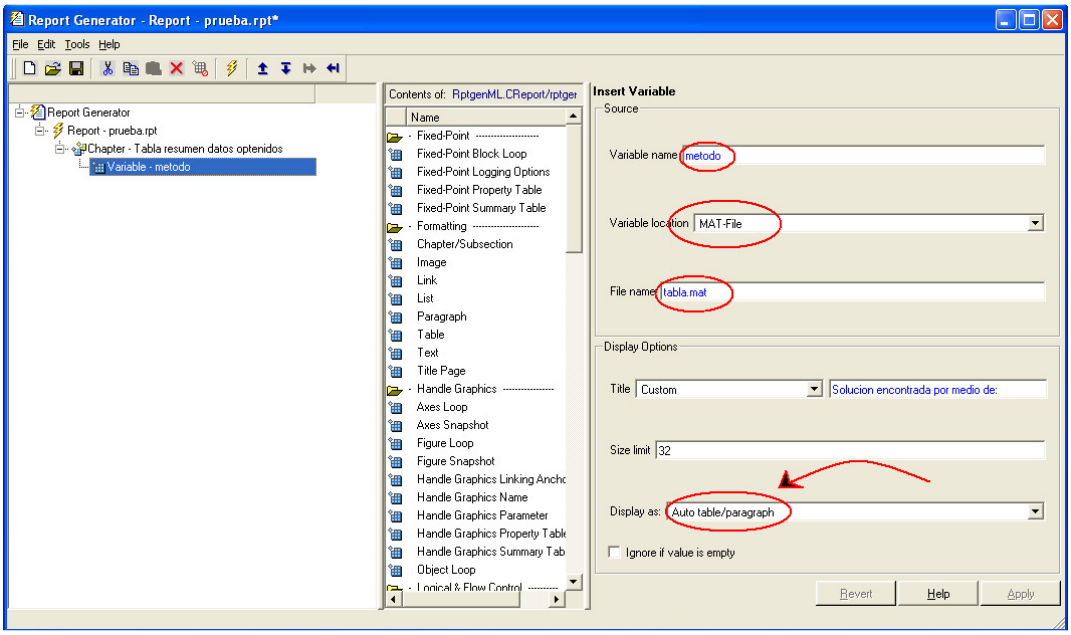


Ilustración 74. Creación de las variables tipo

Se observa la obtención de la variable por medio del archivo tabla.mat y la variable seleccionada “método”.

En la última casilla marcada con un flecha, se indica cómo se quiere mostrar la variable, he aquí el primer problema que se puede tener cuando se trata de la creación de ficheros enriquecidos, como es el caso, ya que al seleccionar que queremos ver el titulo como “Inline test” pese a la correcta generación del documento, se tiene por pantalla el mensaje:

prueba2.xml:5:15:E: document type does not allow element "emphasis" here; missing one of "remark", "titleabbrev", "subtitle", "synopsis", "literallayout", "programlisting", "screen", "para", "simpara", "bridgehead" start-tag

prueba2.xml:5:61:E: character data is not allowed here

Dado el error anterior, la solución, tal y como se tiene en la figura, fue seleccionar auto tabla/paragraph como tipo del título. En este caso, no se daba ningún tipo de mensaje erróneo, pero se obtiene lo siguiente tras la generación del reporte:

Solucion encontrada por medio de:. Uso de operaciones matriciales

(Posteriormente se optó por cambiar “:.” Por “..”

En cuarto lugar se introdujo el tipo de transformada que se usa para el análisis, el desarrollo seguido para la impresión del mismo, en el reporte fue tal cual se realizó el paso anterior. Se muestra en la ilustración73 la elección de la variable.

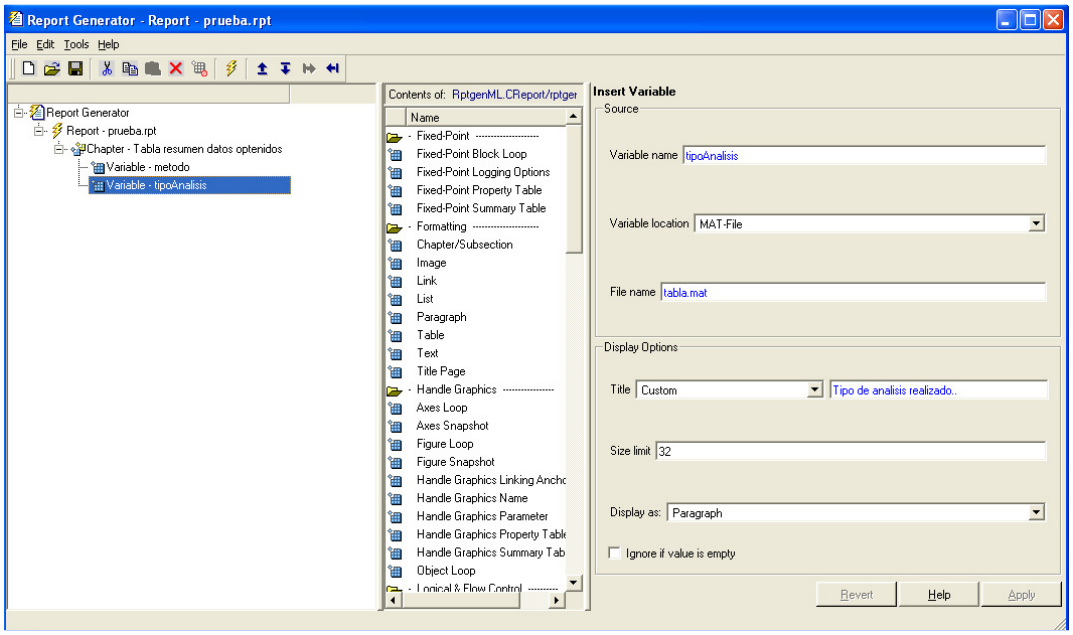


Ilustración 75. Selección de la variable "tipo de análisis"

En quinto lugar, se va a mostrar cómo realizar las tablas donde se muestran las principales características del análisis desarrollado.

En este proceso, se han tenido diversos problemas, ya que como se comento en el punto anterior, no es posible plasmar testo antes de una variable (en este caso variables que se presentarán en forma de tablas)

El problema viene cuando se quiere imprimir un titulo para la tabla.

La solución final, ha sido poner una variable vacía, y ponerle a esta el título que se le quiere poner a la variable siguiente.

Esto es debido a que el titulo se quiere poner antecediendo a la variable (en la línea superior).

Por último se selecciona la variable que se quiere imprimir, en este primer caso el número de niveles que se analizan.

Este proceso es seguido de forma similar para mostrar el número de impulsos que se detecta en cada nivel, y la correlación que existe entre la señal recuperada y la señal inicial.

En la ilustración74 y 75 se muestran los dos pasos descritos, tanto la introducción como la variable vacía, como la variable a imprimir.

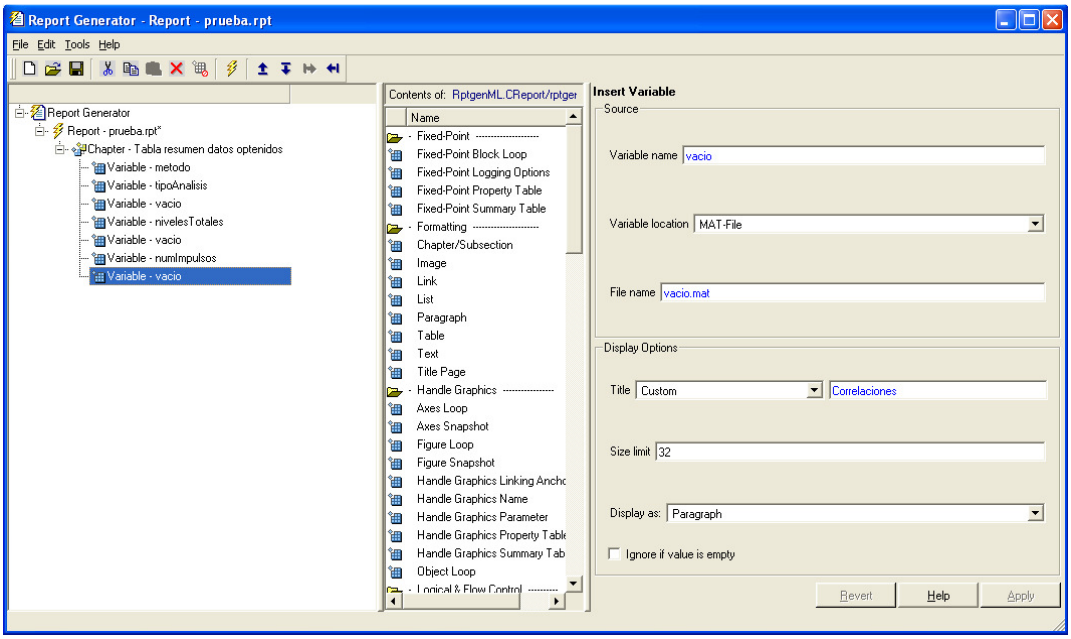


Ilustración 76. Inclusión de la variable vacío para imprimir el título

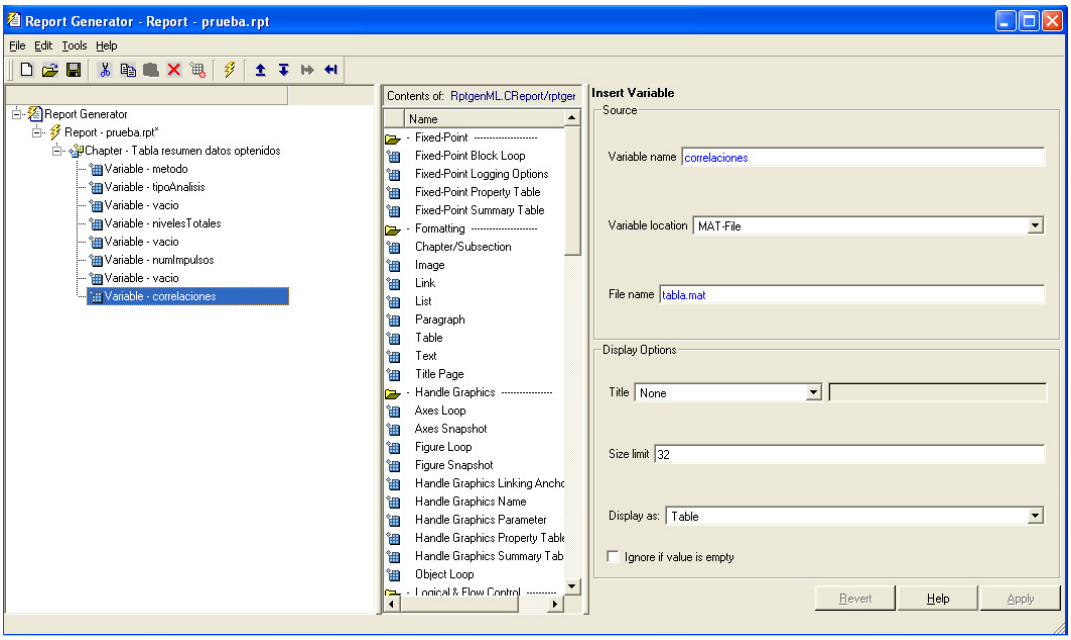


Ilustración 77. Inclusión de la variable correlaciones

✓ Parámetros de salida

Una vez terminado este primer capítulo, se muestra en la ilustración76 un ejemplo de salida del reporte.

Capítulo 1. Tabla resumen datos optenidos

Solucion encontrada por medio de:. Uso de operaciones matriciales

Tipo de analisis realizado.. haar

Niveles.

1	2	3	4

Numero de impulsos.

6	4	5	6

Correlaciones.

1	0.94103	0.90063	0.70197

Ilustración 78. Salida de reporte, primer capítulo.

Como se puede comprobar, se obtienen las características detalladas anteriormente,

En la primera y segunda línea, donde se muestra el método y el tipo de transformada, se ve como en el primer caso se muestra :. Y en el segundo ..

Este problema ya fue comentado, para la impresión final se ha optado por la impresión de ..

✓ Seguimiento de la interfaz, pasos siguientes.

Para el segundo capítulo se mostrará, toda vez que se hayan detectado los 6 impulsos, las características de salida de los mismos, necesarias para detectar el tipo de avería que existe.

En este caso, no siempre tendremos los 6 impulsos, luego hace falta diferenciar este caso.

Para ello se optó por la simplicidad que aporta el hecho, de que en un primer momento, se le asigne a la variable “energiaDeCoeficientes” y “tiempoEntreImpulsos” el string “No se han detectado los 6 impulsos necesarios”. De esta forma, siempre se imprimen, pero en el caso de que no hayan sido detectados, se muestra al usuario que no se generan en el reporte, porque para ello deben haber sido detectados 6.

En un primer lugar no se opto por esta solución, sin embargo, como se ha dicho, es la que aporta mayor simplicidad y claridad a la vez, ya que en caso de elegir no imprimir nada si la variable está vacía tenemos dos inconvenientes:

✓ Aparecerán los títulos del capítulo y las variables, pero no habrá datos.

✓ El usuario puede preguntarse por qué no se muestran las características de los impulsos, si no se muestran, y de esta manera queda clara la razón.

Por último, antes de pasar a mostrar cómo ha sido realizado este apartado, destacar que se muestran las características de los 6 impulsos pertenecientes a la detección de cada nivel por orden de detección, es decir, si se detectan en el primer y en el tercer nivel 6 impulsos, primero se muestras las características de los impulsos detectados en el primer nivel, y después las del tercero.

✓ Operaciones implicadas (capitulo 2)

Se muestra la solución global, ya que los componentes a implementar son muy parecidos a los que teníamos en el capitulo anterior.

En primer lugar el capitulo, en segundo lugar el tipo de método y de transformada, y por último los vectores o matrices según se detecten 6 impulsos una o más veces respectivamente.

En este último caso, para diferenciarlo, se comprobará si ha habido una nueva detección de los impulsos, para incrementar la las variables donde se almacenan la energía y el tiempo entre impulsos.

Este segundo capítulo queda por tanto como se muestra en la ilustración77.

Y genera a la salida los datos mostrados en la ilustración78.

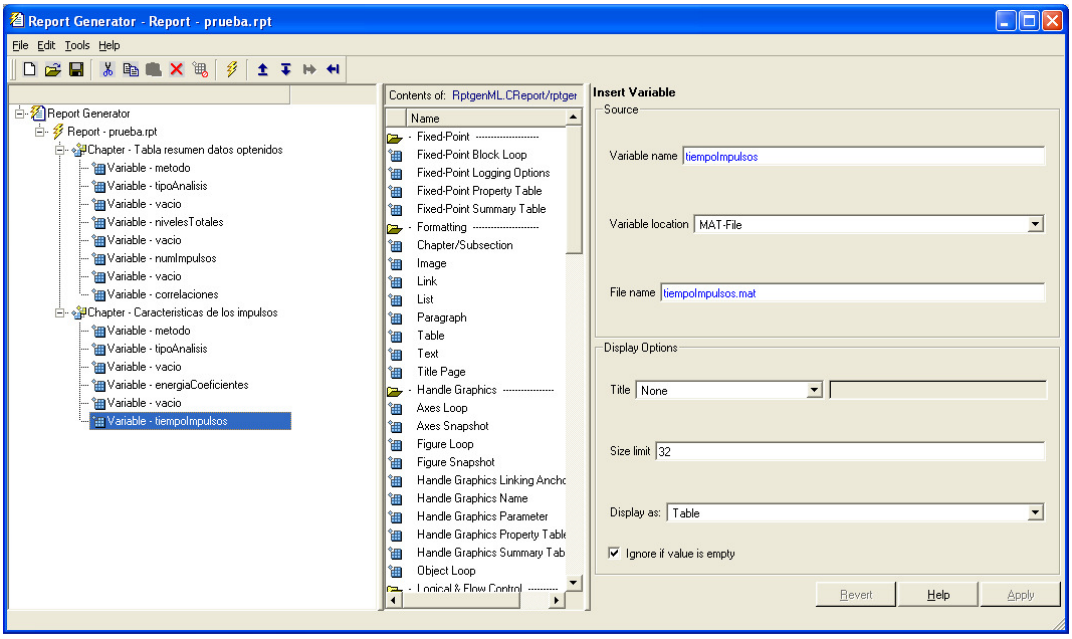


Ilustración 79. Diseño final con la herramienta report generator

- ✓ Parámetros de salida

Capítulo 2. Caracteristicas de los impulsos

Solucion encontrada por medio de.. Uso de operaciones matriciales

Tipo de analisis realizado.. haar

Energia de los coeficientes.

6.8532e-006	0.10267	0.040326	4.9694e-005	2.1114e-006	1.2746e-005
6.8532e-006	0.10267	0.040326	4.9694e-005	2.1114e-006	1.2746e-005

Tiempo entre impulsos.

0.001665	0.37213	0.12155	0.0041625	0.001665
0.001665	0.37213	0.12155	0.0041625	0.001665

Ilustración 80. Salida del reporte, segundo capítulo

Finalmente se puede observar en la ilustración79, un ejemplo de generación de reporte, cuando no hay detección de los 6 impulsos.

Capítulo 2. Caracteristicas de los impulsos

Solucion encontrada por medio de.. Uso de operaciones matriciales

Tipo de analisis realizado.. coiflets

Energia de los coeficientes.

No se han detectado los 6 impulsos necesarios

Tiempo entre impulsos.

No se han detectado los 6 impulsos necesarios

Ilustración 81. Salida del reporte, capitulo 2, no hay detección de los 6 impulsos

Durante la generación del reporte, se han incluido otros dos pasos, que permiten:

- ✓ Eliminar los documentos .doc generados anteriormente.
- ✓ Sacar por pantalla el documento generado. (En un principio se abría nada más se generaba el reporte, pero en la última versión se da la opción al usuario de abrirlo o no.)

En lo que se refiere a estos dos casos, se muestran dos mensajes para que el usuario decida, lo cual se muestra en las figuras 80 y 81.

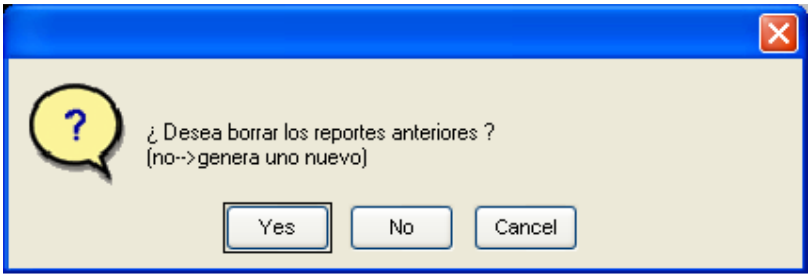


Ilustración 82. Mensaje borrado de reportes anteriores

En este primer caso, si el usuario pulsa yes, se eliminaran los documentos generados anteriormente mediantes las instrucciones:

```
delete([dirActual '\*.rtf']);
delete([dirActual '\*.doc']);
delete([dirActual '\*.XML']);
```

El segundo paso no es tan obvio. Tras eliminar la pestaña en el programa report generator, donde pone que si se quiere generar el reporte después de la generación de mismo (mostrado en la ilustración78), se pregunta al usuario si se quiere abrir el reporte. En este caso se ejecuta el comando:

```
open(ficheroNuevo);
```

Ya que el reporte se genera automáticamente con la extensión “.rpt” , matlab usa el editor por defecto para abrir el archivo, lo cual no permite visualizar correctamente el contenido.

Es por ello que antes de mostrar el reporte se genera una copia con extensión “.doc”, de esta forma matlab usa para abrir el fichero el programa Word, y lo mismo sucedería si se quisiera usar el fichero en formato “.pdf”.

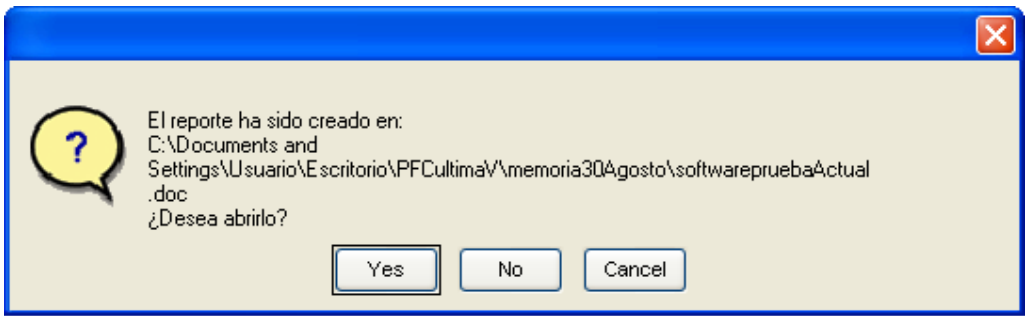


Ilustración 83. Mensaje para abrir el reporte generado

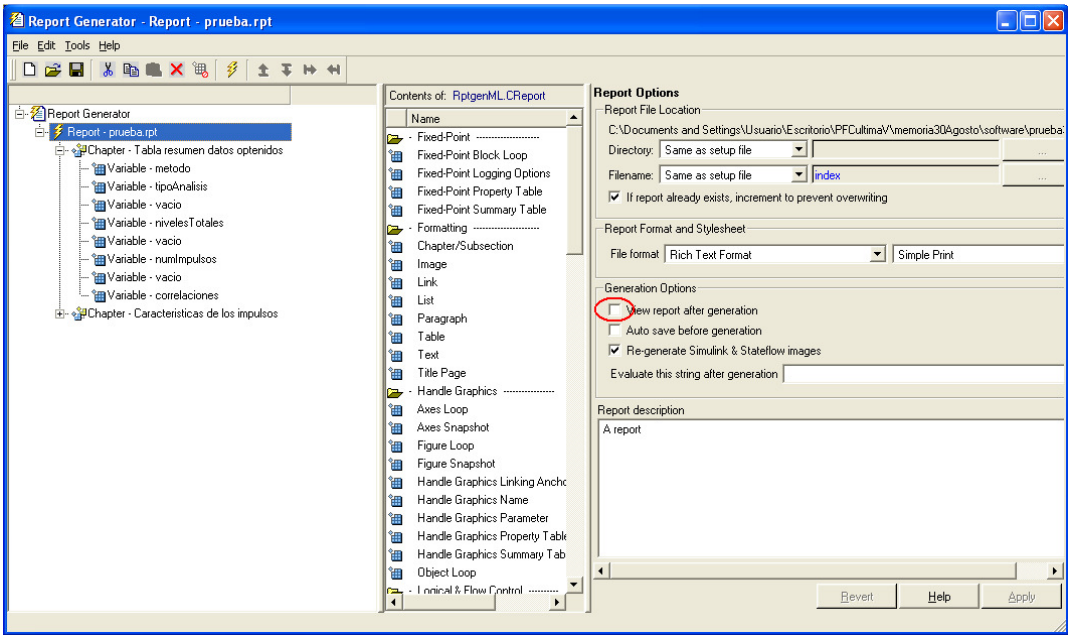


Ilustración 84. Des elección vista del reporte tras generación del mismo

✓ Seguimiento de la interfaz, pasos siguientes.

A continuación se tratarán conjuntamente, los menús de la parte superior de la interfaz, lo que hace referencia a la barra de menús, la barra de estado y la etiqueta que indica el estado del proceso.

5.4.6. Acciones entrada salida: Barra de menús e indicadores de estado.

✓ Cómo introducir este parámetro

En el apartado uno, se vio como introducir la señal de entrada por medio de file->Load.

En este caso nos centramos en primer lugar, del salvado de los coeficientes.

Si se quieren salvar, se debe marcar en el menú options, la casilla, save coeficientes.

Una vez está marcada, se habilita el salvado de los coeficientes, siempre y cuando se hayan obtenido por medio del análisis. Es por tanto, que en caso de que no hayan sido obtenidos los coeficientes por medio del análisis pertinente, seguirá sin habilitarse la pestaña "coeficientes".

En la ilustración83 se muestra la selección de la opción de salvar los coeficientes, y en la ilustración84 se muestra la selección de salvar los coeficientes en un archivo .doc

Ya que se ha tenido problemas a la hora de abrir el archivo .pdf se ha desestimado la opción de usar este método, con lo que permanecerá deshabilitado bajo cualquier concepto.

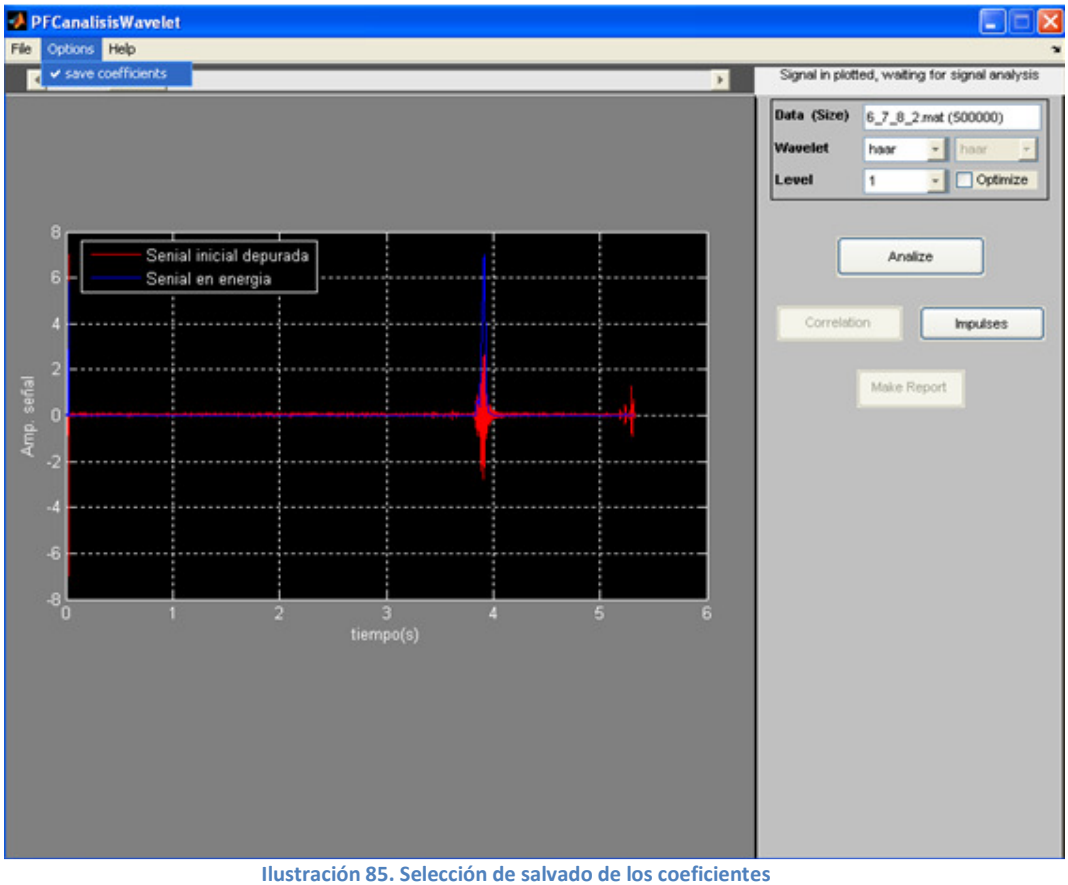


Ilustración 85. Selección de salvado de los coeficientes

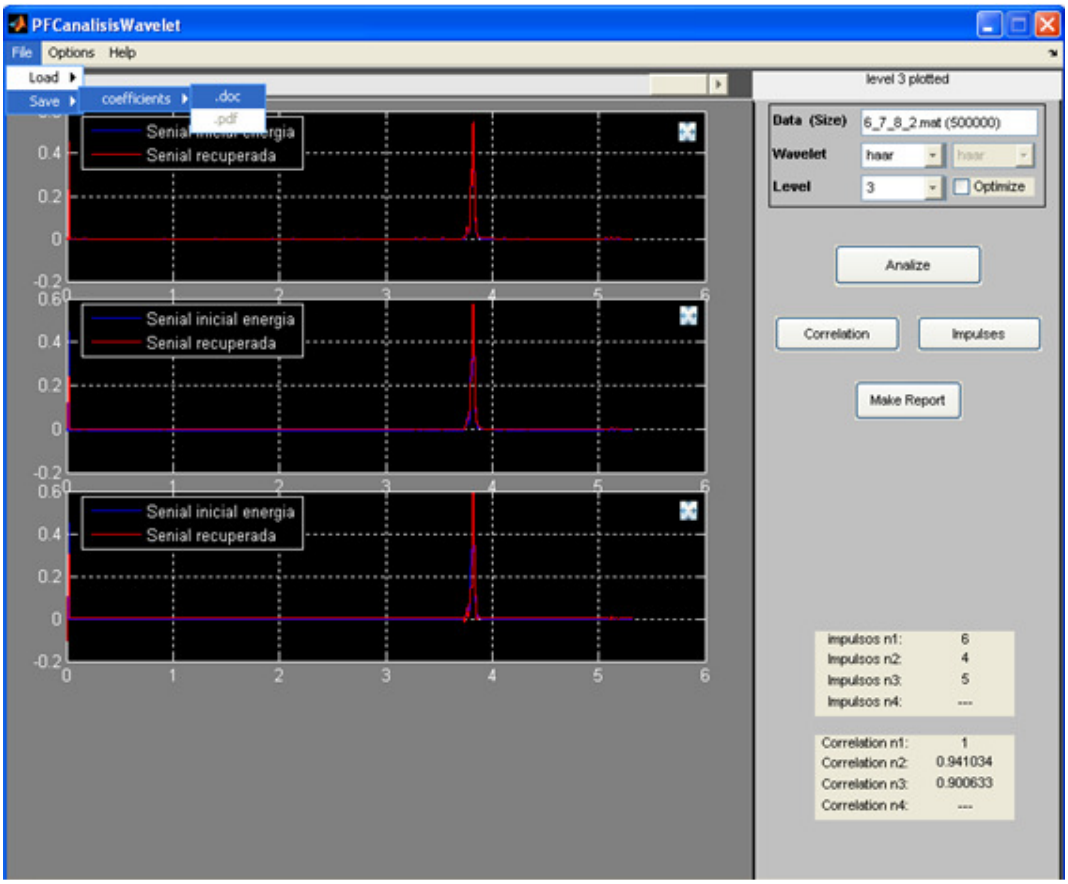


Ilustración 86. Salvar los coeficientes a archivo .doc

✓ Operaciones implicadas

Antes de reseñar las operaciones principales que se llevan a cabo, es importante tener en cuenta que los botones pertenecientes a los menús, son implementados de una manera diferente.

Para agregar un menú a la interfaz, se debe de hacer uso del menú editor de matlab, lo cual se presenta en la ilustración85.

Tras este menú, se despliega una pantalla donde se permite configurar el diseño.

Se agregan por lo tanto 3 botones, File, Option y Help respectivamente.

A su vez estos botones permiten el acceso a otras acciones. En la ilustración100 vemos las características principales que deben especificarse.

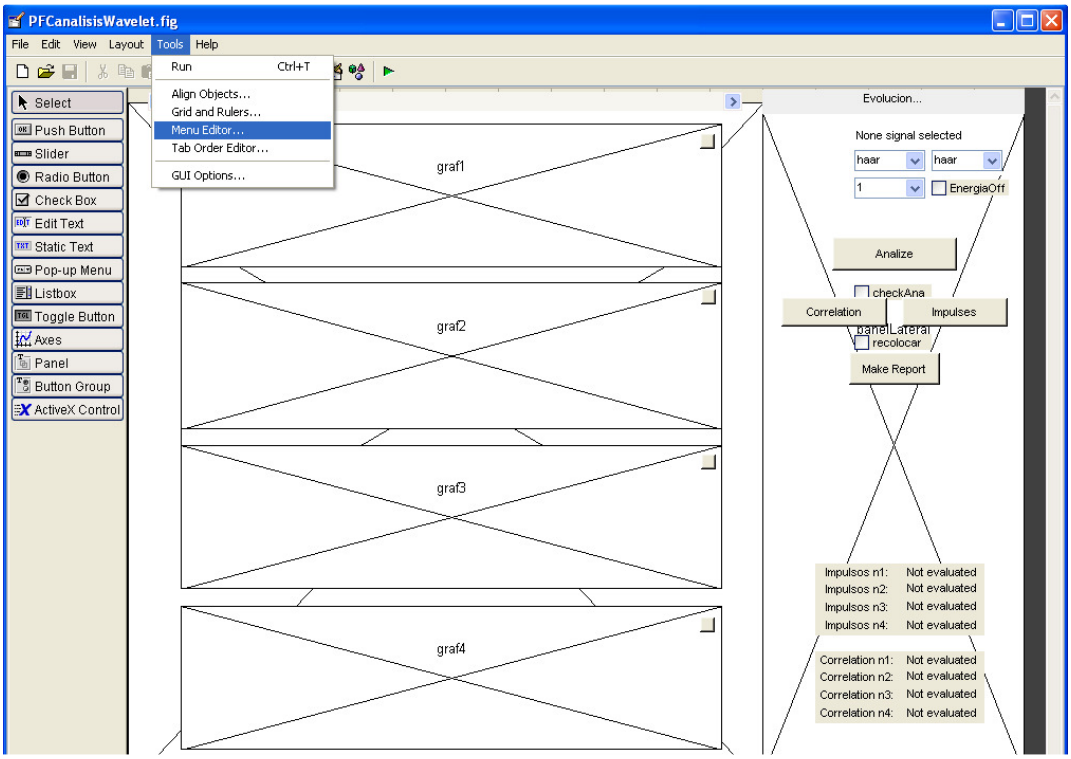


Ilustración 87. Selección de menú editor

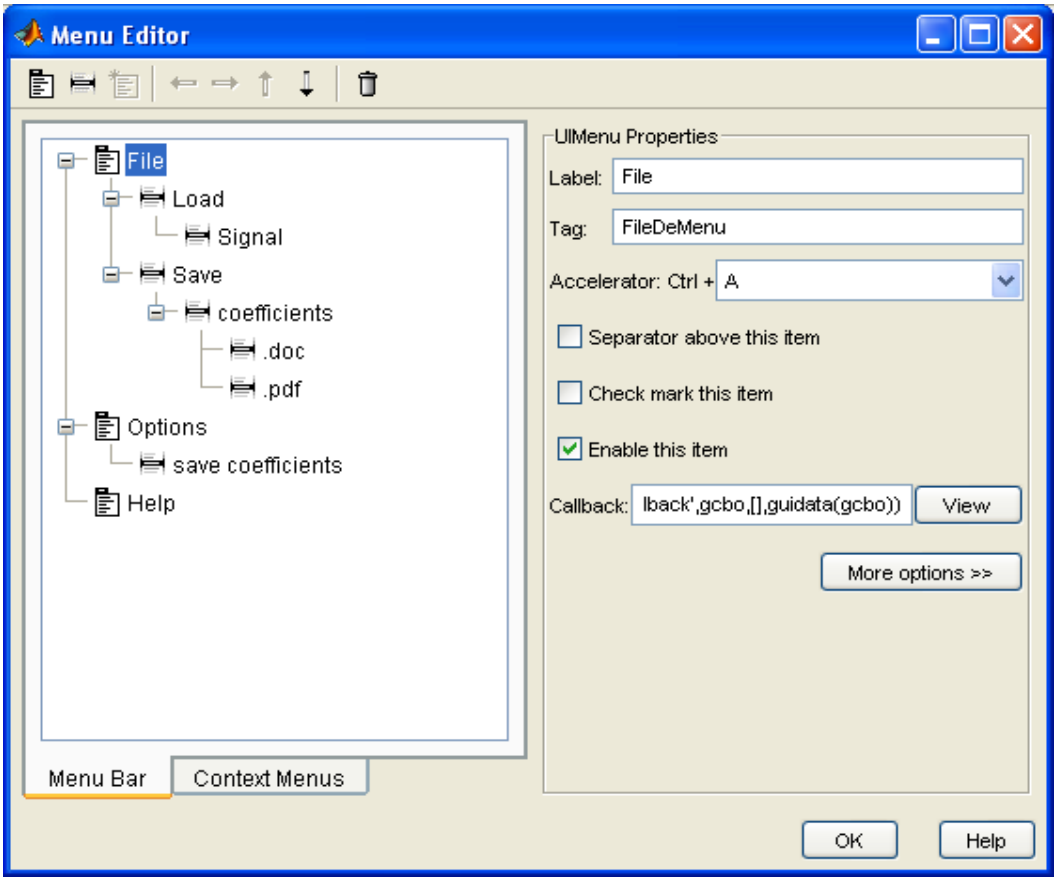


Ilustración 88. Menú editor

Como se muestra en la ilustración100, está perfectamente esquematizado, lo cual permite hacerse una idea de cómo se desplegaran los menús.

En este caso se insertan las etiquetas y se indica el "callback", o dicho de otra manera, la función a la que llamaran cuando se pulse.

Existen dos opciones implicadas en cuanto a la barra de los menús se refiere, la primera en la ilustración97, habilitar la barra de salvado de los coeficientes en documentos, y el segundo paso en la ilustración98 que conlleva la generación del reporte con los coeficientes pertinentes.

Este paso tiene gran similitud, en cuanto al desarrollo, el apartado anterior donde se describía la herramienta "report generator".

Se muestra en la ilustración87 los pasos que se realizaron para la generación del reporte.

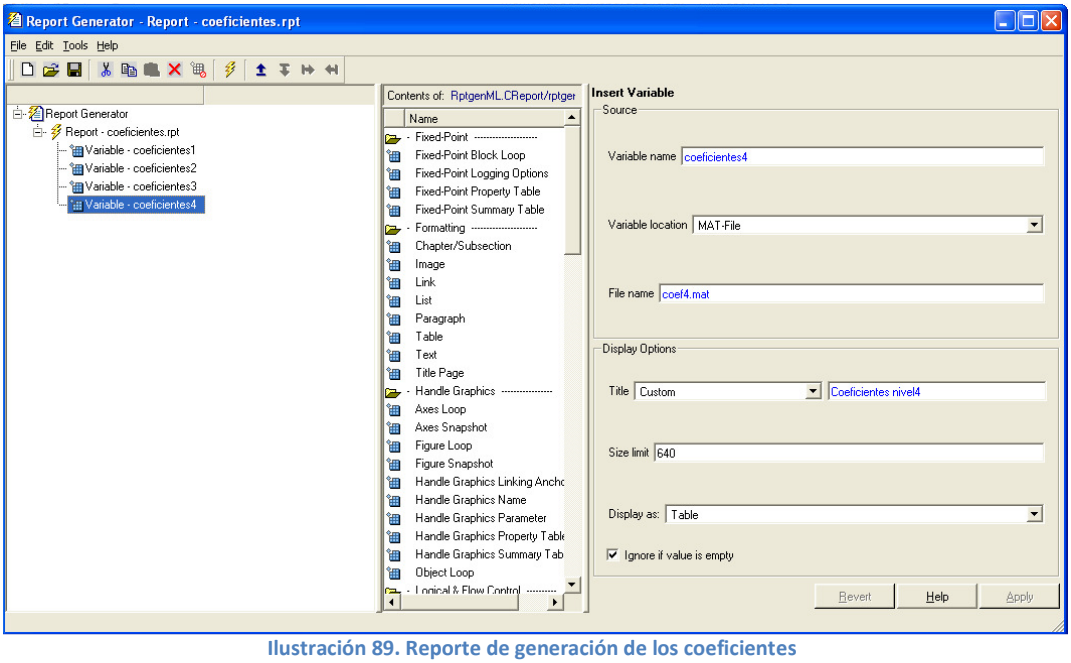


Ilustración 89. Reporte de generación de los coeficientes

A destacar, que los coeficientes pueden llegar a ser 640, luego se debe aumentar el tamaño de la variable para que pueda imprimirse.

Además si la variable está vacía (no se ha generado porque no se han obtenido en el análisis) no se muestra, como se marca en la casilla de la parte inferior derecha.

- ✓ Parámetros de salida

En este caso, cuando se pulsa el botón de salvado de los coeficientes al archivo .doc se genera el reporte y se pregunta si se quiere abrir el archivo.

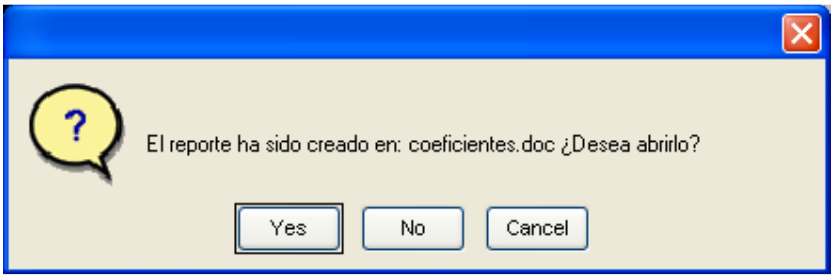


Ilustración 90. Pregunta para abrir reporte de los coeficientes

Dado que en la generación del reporte se imprimen los 640 coeficientes, no se muestra la totalidad del reporte en la ilustración101.

Por último, comentar que cuando se analiza la señal mediante funciones matlab, no se da la opción de guardar los coeficientes, ya que son 1280 coeficientes y no aportan tanto como en el caso de las funciones matriciales. Para ello se deshabilita la opción de salvado.

Coeficientes nivell. [-0.12345 ; 0.74576 ; -0.012085 ; -0.011845 ; -0.012232 ; -0.012414 ; -0.012408 ;
-0.012424 ; -0.012423 ; -0.012151 ; -0.012221 ; -0.012306 ; -0.012438 ; -0.012329 ; -0.012422 ;
-0.012393 ; -0.01239 ; -0.01238 ; -0.012404 ; -0.012365 ; -0.012423 ; -0.01243 ; -0.012442 ; -0.012429 ;
-0.012401 ; -0.012384 ; -0.012425 ; -0.012472 ; -0.012378 ; -0.012447 ; -0.012448 ; -0.012418 ;
-0.012469 ; -0.012381 ; -0.012332 ; -0.012323 ; -0.012283 ; -0.012338 ; -0.012396 ; -0.012437 ;
-0.012439 ; -0.012437 ; -0.012477 ; -0.012419 ; -0.012385 ; -0.012372 ; -0.012458 ; -0.012443 ;
-0.012478 ; -0.012419 ; -0.012473 ; -0.012451 ; -0.012396 ; -0.012406 ; -0.012433 ; -0.012269 ;
-0.012444 ; -0.012404 ; -0.012427 ; -0.012416 ; -0.012428 ; -0.01243 ; -0.012413 ; -0.012389 ;
-0.012396 ; -0.012325 ; -0.012435 ; -0.01244 ; -0.012406 ; -0.012429 ; -0.012397 ; -0.012421 ; -0.01243
; -0.012377 ; -0.012399 ; -0.012421 ; -0.012402 ; -0.012413 ; -0.01247 ; -0.012438 ; -0.01243 ;
-0.012458 ; -0.012357 ; -0.012364 ; -0.012345 ; -0.012322 ; -0.012374 ; -0.012432 ; -0.012399 ;
-0.012377 ; -0.012398 ; -0.012463 ; -0.012382 ; -0.012379 ; -0.012396 ; -0.012409 ; -0.012316 ;
-0.012436 ; -0.012409 ; -0.012395 ; -0.01241 ; -0.012386 ; -0.012402 ; -0.012351 ; -0.012426 ;
-0.012401 ; -0.01233 ; -0.012455 ; -0.012388 ; -0.012422 ; -0.012362 ; -0.012433 ; -0.012324 ;
-0.012412 ; -0.012431 ; -0.012369 ; -0.012323 ; -0.012391 ; -0.012417 ; -0.012396 ; -0.012438 ;
-0.012446 ; -0.012342 ; -0.012303 ; -0.012352 ; -0.01241 ; -0.012375 ; -0.012263 ; -0.012417 ;
-0.012429 ; -0.0124 ; -0.012384 ; -0.012336 ; -0.012369 ; -0.012306 ; -0.012395 ; -0.012427 ; -0.012332
; -0.012373 ; -0.012436 ; -0.01239 ; -0.012423 ; -0.012311 ; -0.012359 ; -0.012367 ; -0.012347 ;
-0.012366 ; -0.01241 ; -0.012382 ; -0.012352 ; -0.01238 ; -0.01242 ; -0.012417 ; -0.01239 ; -0.012343 ;

Ilustración 91. Reporte de los coeficientes de salida

✓ Seguimiento de la interfaz, pasos siguientes.

Una vez se han descrito los menús, sólo queda comentar la barra de proceso y la etiqueta que muestra en que paso nos hallamos,

Dada la simplicidad de ambos elementos, no se abrirá un apartado especial para ambos.

Tan solo es necesario comentar, que tanto la barra de estado como el mensaje a imprimir, el cual indica el estado por el que encuentra el programa, son pasados como parámetro, cada vez que se llama a la interfaz desde el programa de cómputo de los coeficientes.

Por tanto si se está evaluando un segundo nivel de cuatro que se requirieron, asigna un porcentaje a la barra, y del mismo modo el mensaje a mostrar.

Es la interfaz, una vez tiene estos parámetros, la que los muestra por pantalla en forma de ambos objetos.

5.4.7. Manual de la interfaz: botón help del menú.

Tras el minucioso análisis de cada uno de los apartados que componen la interfaz, sólo queda destacar la opción de ayuda mostrada en el esquema de la ilustración100 en último lugar. Se desplegará por tanto, un manual realizado a tal efecto.

Ya que este manual se desplegará llamando a un fichero aparte (help.doc) se adjunta por tanto en la carpeta software del proyecto donde están las funciones pertenecientes a la interfaz.

En este apartado se incluye el manual de forma miniaturizada en la ilustración90:

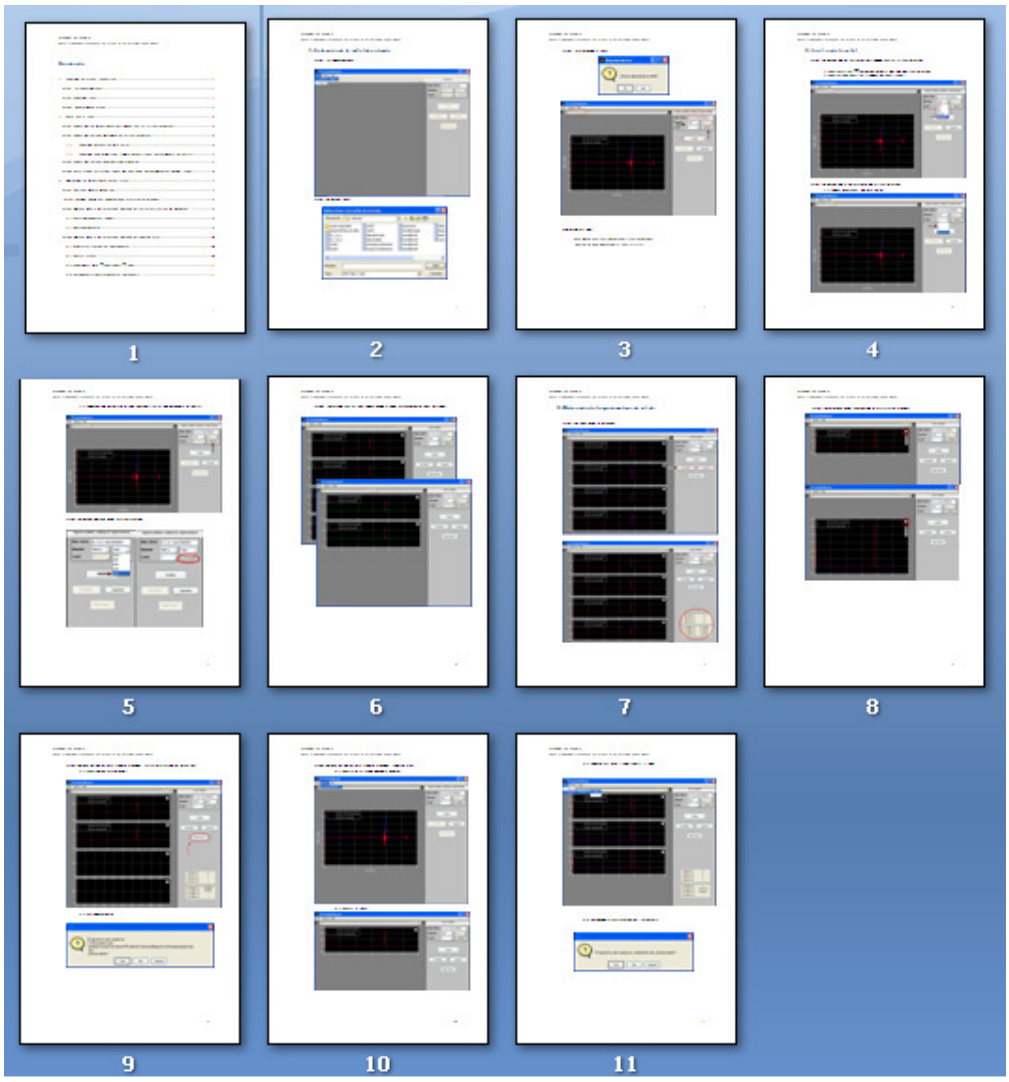


Ilustración 92. Manual de la interfaz

CAPITULO 6. CONCLUSIÓN

En este apartado, se replica el punto 4.3. a modo de conclusión final, y para que ejerza de capitulo introductorio para el punto 7, donde se establecen las líneas generales para futuros estudios y mejoras.

Los resultados y los razonamientos anteriores, aportan una visión global del problema que permite destacar las ventajas e inconvenientes de cada método, así como la conclusión de cuál debe ser el procedimiento a seguir para futuros estudios.

En la ilustración91 se muestra a grandes rasgos el análisis global del proyecto, para a continuación detallar cada método.

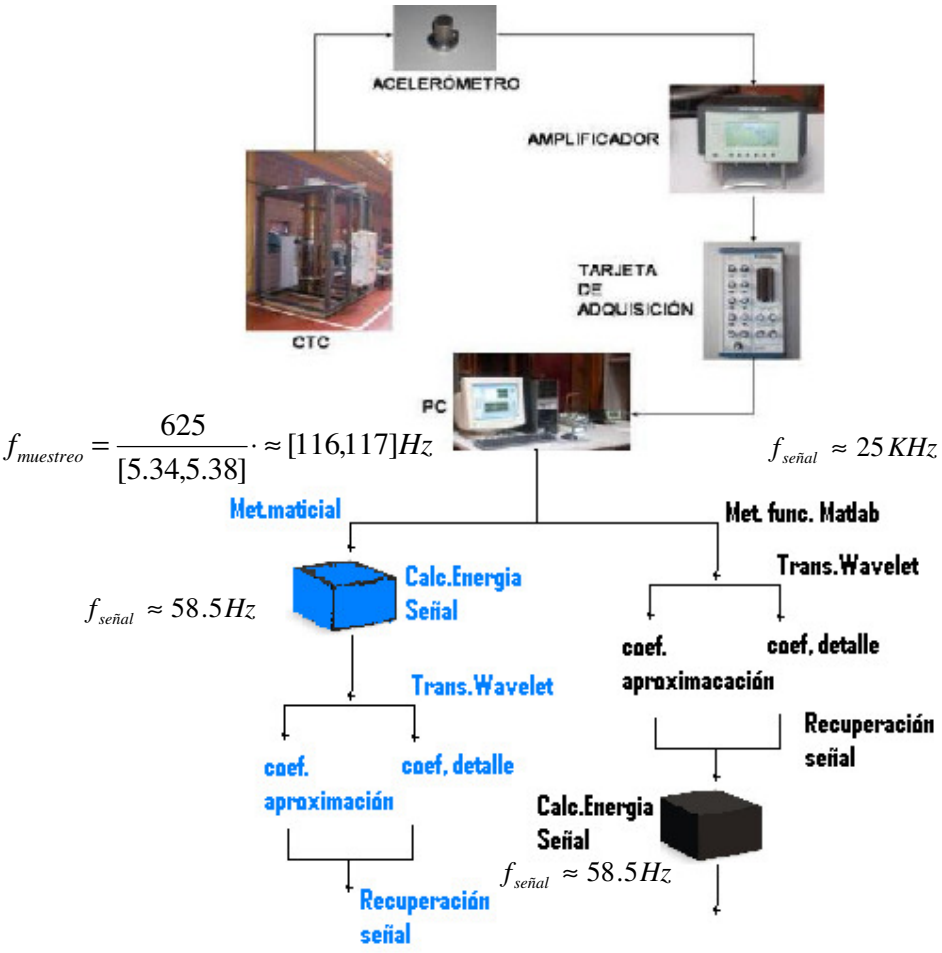


Ilustración 93. Resumen gráfico de ambos métodos

6.1. ANÁLISIS CON MÉTODOS MATRICIALES

Ventajas:

- ✓ Análisis transparente con funciones implementadas.
- ✓ Se trabaja con señales optimizadas en cuanto número de muestras. (Mínimo número de muestras para detectar 6 impulsos)
- ✓ Obtención de la señal con un número de coeficientes de wavelet bajo (640 coeficientes),además, se tiene la matriz capaz de recomponer la señal, luego sólo es necesario la realización de una simple operación para recuperar la señal de entrada.
- ✓ Método optimizado para la selección de coeficientes.
- ✓ Correcta detección de los impulsos en un alto porcentaje.
- ✓ Menor peso computacional que el análisis con funciones wavelet.

Desventajas:

- ✓ No se puede trabajar con señales con gran número de muestras (debido a las matrices que se componen con la misma dimensión)
- ✓ No se puede realizar un análisis frecuencial detallado
- ✓ En algunos casos la detección es mas imprecisa que en el caso de hacer uso de las funciones de matlab.

6.2. ANALISIS DEL MÉTODO QUE HACE USO DE FUNCIONES MATLAB.

Es evidente que las ventajas y desventajas del método anterior se verán contrapuestas en este apartado, aún así se reseñan para una más completa explicación.

Ventajas:

- ✓ Posibilidad de hacer un análisis frecuencial exhaustivo: al trabajar con la señal en tiempo, se puede aplicar la transformada directamente, obteniendo un filtrado paso alto y paso bajo en cada nivel, y con los coeficientes, ser capaz de recuperar las frecuencias que interesen.
- ✓ Fiabilidad a la hora de obtener los impulsos correctos, ya que la señal filtrada, habiéndose eliminado impulsos espurios, permite una mejor obtención de la energía de los impulsos (eliminación de ruido).

Desventajas:

- ✓ Mayor coste computacional, al trabajar con señales con mayor número de muestras (5000000).
- ✓ Opacidad de las funciones de matlab que se usan.

- ✓ Los coeficientes que se obtienen son mucho mayores en número, siendo necesario, además, las funciones propias de matlab para la recuperación de la señal.

6.3. CONCLUSIÓN FINAL.

Es inevitable destacar la necesidad de ambos métodos para recapacitar acerca del correcto análisis y la correcta extracción de los parámetros de diagnostico.

Una vez detalladas las ventajas e inconvenientes, se plantea cual es la línea ideal a seguir, partiendo de la base de que se han alcanzado los objetivos planteados al inicio.

Para ambos métodos se tiene una correcta detección, aunque es cierto que en el caso de la señal 6_7_8_2.mat los métodos mostraban una ligera variación en el primer impulso.

Pues bien, toda vez, se tienen implementado ambos, se concluye que es necesario realizar el análisis por ambos métodos para detectar si los datos obtenidos, son los idóneos para la clasificación de las posibles averías que se dan en el transformador, y en el caso, de que no se tengan exactamente los mismos resultados con ambos métodos, usar los datos extraídos con el segundo de ellos.

Por tanto, el método matricial por sí sólo, es una herramienta que enmascara las perturbaciones en frecuencia de la señal en forma de energía de la misma, luego puede hacer que en ciertos casos (señal 6_7_8_2.mat) la clasificación sea errónea.

Al final del documento, tras la explicación de cómo se ha realizado la interfaz de usuario, se muestra un apartado donde se aconsejan nuevas ampliaciones a este estudio y se detalla una vía paralela para llegar a los resultados anteriores.

CAPITULO 7. TRABAJOS FUTUROS.

Teniendo cumplidos los objetivos marcados al inicio, se va a resaltar en primer lugar, una posible mejora para la detección de los parámetros de diagnostico.

Dados ambos métodos, el objetivo será la obtención de un análisis detallado de las frecuencias a las que pertenece cada impulso, ya que con la transformada de wavelet, se es capaz de desmenuzar la señal en tiempo y frecuencia. Para ello, basta con incrementar los niveles de análisis de la transformada implementada con las funciones de matlab.

Aunque sería suficiente tener un análisis frecuencial, se puede establecer un método menos exhaustivo.

Se realiza el análisis hasta donde el filtrado en frecuencias no permita la detección de los 6 impulsos y se toma el nivel superior a este (donde si se detectaban) a modo de fijar un rango de frecuencias límite para el filtrado previo al análisis de la señal.

De igual forma se pueden ampliar los tipo de transformadas, así como intentar la aplicación de este programa (representación frecuencial en 3D), para otro tipo de señales no repetibles en el tiempo, como es el caso de la señales procedentes de ECG.

Una mejora en la selección de los coeficientes de la transformada (a la que se hace referencia en el capítulo 3.1.2. y que no se llego a implementar) es una vez se tiene la señal recuperada, tratar de obviar frecuencias intermedias que provienen de coeficientes en detalle de niveles intermedios, es decir, tras el algoritmo de búsqueda, desde el último, hasta el primer nivel analizado, se pasa a realizar el algoritmo en sentido inverso (iniciando desde el primer nivel desde donde se ha sido capaz de recuperar la señal) hasta el último

BIBLIOGRAFÍA

- **[1] Tratamiento de señales digitales mediante wavelets y su uso con Matlab**.
  Felix Martinez Giménez/Alfredo Peris Manguillot/Francisco Rodenas Escribá
  Editorial Club Universitario, Valencia, 2004

- **[2] Wavelet tour of signal processing**
  Stéphane Mallat
  Editorial Academic Press, 1998
- **[3] Tutorial introductorio a la teoria de Wavelet**
  Samir Kouro/Rodrigo Musalem M.
  Paper 2002

- **[4] Manual de interfaz gráfica de usuario Matlab**
  Diego Orlando Barragán Guerrero

- **[5] Detección de averías en cambiadores de tomas en carga de trasformadores basado en el patrón de vibraciones.**
  Edwin Rivas Trujillo
  Tesis Doctoral, 2009

- **[6] Wavelet Methods for Elliptic Partial Differential Equations**
  Karsten Urban
  Editorial Series, 2009

- **[7] Transient signal analysis and classification for condition monitoring of power switching equipment using wavelet transform and artificial neural networks**
  P. Kang, D. Birtwhistle and K. Khouzam, 1998

- **[8] "Ten Lectures on Wavelets". The Society for Industrial and Applied Mathematics**
  I. Daubechies, 1992

ANEXO A: Presupuesto

En este anexo se detalla un presupuesto detallado, con cada uno de los costes de desarrollo del proyecto.

Se tiene en cuenta para ello, los recursos utilizados así como las tareas en las que se ha dividido.

A.1. Recursos

En primer lugar, se evalúa el coste de los recursos utilizados a lo largo del proyecto.

A.1.1. Recursos Materiales

Ya que la totalidad del proyecto se ha realizado con un ordenador personal, dadas las características del mismo:

Procesador: Intel Core 2 Duo CPU 2GHz
Memoria: 2 GHz

Se tasa en 700€

Dado que el ordenador queda obsoleto en 3 años y el proyecto ha tenido una duración aproximada de 1 año y 1 mes, los costes totales ascienden a **253€**

A.1.2. Recursos Software

Los programas usados, y el coste de sus licencias son los siguientes:

Nombre del software	Coste
Windows XP Home Edition	150€
Microsoft Office 2007	500€
Matlab 7.0	6000€

Tabla 11. Software utilizado para el proyecto

Luego suponiendo una vida útil de 3 años, los costes totales ascienden a **2414€.**

A.1.3. Recursos Humanos.

Se han considerado los siguientes costes para un ingeniero de Telecomunicación:

- Coste de ejecución por hora normal: 10€
- Coste de ejecución por hora extra: 15€

Dado que la duración del proyecto ha sido de 11 meses, en los cuales durante 3 meses se le ha dedicado una jornada completa de 8 horas (hora normal), y durante los 8 restantes, se le ha dedicado media jornada, 4 horas (extras) tenemos. A los 3 meses de jornada completa, tenemos que ponerle 6 horas de media los fines de semana que se catalogan como extras (8 días al mes).

$$(8\cdot 22\cdot 3)horas\cdot 10\frac{euros}{hora}+(6\cdot 8\cdot 3)horas\cdot 15\frac{euros}{hora}+(4\cdot 8\cdot 30)horas\cdot 15\frac{euros}{hora}=21840€$$

A.2. Resumen costes del proyecto.

Concepto	Coste
Hardware	253€
Software	2414€
Recursos Humanos	21840€
Beneficio (10%)	2451€
Total Sin IVA	26958€
IVA (16%)	4313€
Total con IVA	31271€

Tabla 12.Costes totales del proyecto

Luego el coste del proyecto asciende a:

TREINTA Y UN MIL DOSCIENTOS SETENTA Y UN EUROS

ANEXO B: Código Matlab

analisisBior(senialEnergia,longDep,nivelTope,energia)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Author: José María Lucas Zaragoza
% Date: 22 Augost 2010
% Este metodo es el encargado de hacer el analisis haciendo uso de las
% funciones wavelet de matlab, el tipo de wavelet seleccionado debe ser
% Biorthogonal
% Parametros de entrada:
%   - senialEnergia: senial de entrada a la cual realizar el anÃ¡lisis
%   - longDep: longitud de la seÃ±al depurada
%   - nivelTope: nivel maximo a realizar el analisis
%   - energia: booleano(0 o 1) que indica si la seÃ±al de entrada es en
%     energia o en tiempo
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function senial=analisisBior(senialEnergia,longDep,nivelTope,energia)

% se realiza la transformada para cada uno de los niveles
for nivel=1:nivelTope
   % seleccionar el tipo de transformada
   cadenaNivel=[int2str(nivel) '.' int2str(nivel)];
   wname = ['bior' cadenaNivel];
   % se hallan los coeficientes del filtrada
   [ra,rd] = biorwavf(wname);
   % se obtienen los coeficientes de reconstruccion y de deconstruccion
   [Lo_D,Hi_D,Lo_R,Hi_R] = biorfilt(rd,ra);
   % se toman los coeficientes
   coeficientesAprox = filter(Lo_D,1,senialEnergia);
   coeficientesDetalle = filter(Hi_D,1,senialEnergia);
   % se recalcula la seÃ±al a partir de los coeficientes
   recuperadaAprox=filter(Lo_R,1,coeficientesAprox);
   recuperadaDif=filter(Hi_R,1,coeficientesDetalle);
   % se hallan los impulsos con la seÃ±al en aproximacion y en diferencias
   [impulsosA,numImpulA]=deteccionImpulsosEnergia(recuperadaAprox,longDep);
   [impulsosD,numImpulD]=deteccionImpulsosEnergia(recuperadaDif,longDep);
   coeficientes=[coeficientesAprox];
   % se halla la correlacion de la seÃ±al recuperada respecto la inicial
   correlacion=corr(senialEnergia',recuperadaAprox');
   % ahora se analiza el caso en que la seÃ±al de entrada no venga dada en
   % energia, si no en tiempo.
   if(~energia)
```

```
        % No debemos cambiar el nÃºmero de muestras de la seÃ±al de energÃa
        % cada vez,debemos seleccionar el nÃºmero de muestras al inicio
        muestras=625;
        save muestras.mat muestras
        % se detecta el numero de muestras optimo para detectar los 6
        % impulsos
        detmuestras(senialEnergia);
        % se depura la seÃ±al recuperada en aproximacion anterior (se toma
        % la energia de la senial
        [recuperadaEnergia2,senialDepRecuperada]=depurar(recuperadaAprox');
        % tambien se depura la senial de entrada (que no es en energia),
        % para poder compararla con la recuperada
        [senialEnergia2,senialDepEnergia]=depurar(senialEnergia');
        % longitud de la senial depurada
        longDep=length(senialDepRecuperada);
        % se detectan los impulsos de la senial
        [impulsos2,numImpul2]=deteccionImpulsosEnergia(recuperadaEnergia2,longDep);

        coeficientes2=coeficientes;
        % puede suceder que la senial inicial y la depurada no tengan la
        % misma longitud (tienen el mismo numero de muestras, pero entre el
        % primer y el ultimo impulsos puede que varie un poco.
        dif=length(senialEnergia2)-length(recuperadaEnergia2);
        if(dif<0)
           recuperadaEnergia2=[recuperadaEnergia2(1:end-dif) zeros(1,dif)];
           senialEnergia2=[senialEnergia2 zeros(1,dif)];
        else
           senialEnergia2=[senialEnergia2(1:end-dif) zeros(1,dif)];
           recuperadaEnergia2=[recuperadaEnergia2 zeros(1,dif)];
        end
        % ahora si, se hace la correlacion con las seniales del mismo tamanio
        correlacion2=corr(senialEnergia2',recuperadaEnergia2');
        % se normalizan las seniales para hallar la energia
        recuperadaEnergia2=normalizacion(recuperadaEnergia2')';
        senialEnergia2=normalizacion(senialEnergia2')';
        % se halla siempre que tengamos 6 impulsos
        if(numImpul2==6)
           numCoeficientes=length(coeficientes2);

coeficientes_energia=energyCoeficient(senialEnergia2,impulsos2,numImpul2,longDep);
        end
        % se llama a la interfaz para mostrar los resultados obtenidos
        PFCanalisisWavelet(senialEnergia2,recuperadaEnergia2,
numImpul2,correlacion2,coeficientes2,longDep,nivel);
     % en caso de que la seÃ±al venga en energia, se devuelven directamente
     % los resultados
     else
        recuperadaAprox=normalizacion(recuperadaAprox')';
        senialEnergia=normalizacion(senialEnergia')';
        PFCanalisisWavelet(recuperadaAprox,senialEnergia,
numImpulA,correlacion,coeficientes,longDep,nivel)
     end
```

```
end
```

analisisCoiflets(senialEnergia,longDep,nivelTope,energia)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Author: José María Lucas Zaragoza
% Date: 22 Augost 2010
% Este metodo es el encargado de hacer el analisis haciendo uso de las
% funciones wavelet de matlab, el tipo de wavelet seleccionado debe ser
% Coiflets
% Parametros de entrada:
%   - senialEnergia: senial de entrada a la cual realizar el anÃ¡lisis
%   - longDep: longitud de la seÃ±al depurada
%   - nivelTope: nivel maximo a realizar el analisis
%   - energia: booleano(0 o 1) que indica si la seÃ±al de entrada es en
%     energia o en tiempo
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function senial=analisisCoiflets(senialEnergia,longDep,nivelTope,energia)
%tipo de analisis
tipo='coif';
% se realiza la transformada para cada uno de los niveles
for nivel=1:nivelTope
   % se construyen los parametros para el filtrado
   cadenaNivel=[int2str(nivel)];
   wname = [tipo cadenaNivel];
   % se hallan los coeficientes del filtrada
   [r] = wfilters(wname);
   [LO_D,HI_D,LO_R,HI_R] = orthfilt(r);
   % se obtienen los coeficientes de deconstrucion y construccion
   coeficientes= filter(LO_D,1,senialEnergia);
   % y se obtienen las seÃ±ales tras el filtrado
   recuperada=filter(LO_R,1,coeficientes);
   coeficientesD=filter(HI_D,1,senialEnergia);
   recuperadaD=filter(HI_R,1,coeficientesD);
   % se usan los coeficientes de aproximacion para la reconstruccion
   [impulsos,numImpul]=deteccionImpulsosEnergia(recuperada,longDep);
   % normalizacion de las seÃ±ales
   recuperada=normalizacion(recuperada')';
   senialEnergia=normalizacion(senialEnergia')';
   % se obtienen la correlacion entre la senial recuperada e inicial
   correlacion=corr(senialEnergia',recuperada');
   % si la seÃ±al no esta en energia, se pasa a energia para comparar
   % resultados
   if(~energia)
      % No debemos cambiar el nÃºmero de muestras de la seÃ±al de energÃa
      % cada vez,debemos seleccionar el nÃºmero de muestras al inicio
```

```
        muestras=625;
        save muestras.mat muestras
        % se detecta el numero de muestras optimo para detectar los 6
        % impulsos
        detmuestras(senialEnergia);
        % se depura la seÃ±al recuperada en aproximacion anterior (se toma
        % la energia de la senial
        [recuperadaEnergia2,senialDepRecuperada]=depurar(recuperada');
        % tambien se depura la senial de entrada (que no es en energia),
        % para poder compararla con la recuperada
        [senialEnergia2,senialDepEnergia]=depurar(senialEnergia');
        % longitud de la senial depurada
        longDep=length(senialDepRecuperada);
        % se detectan los impulsos de la senial
        [impulsos2,numImpul2]=deteccionImpulsosEnergia(recuperadaEnergia2,longDep);

        coeficientes2=coeficientes;
        dif=length(senialEnergia2)-length(recuperadaEnergia2);
        % puede suceder que la senial inicial y la depurada no tengan la
        % misma longitud (tienen el mismo numero de muestras, pero entre el
        % primer y el ultimo impulsos puede que varie un poco.
        if(dif<0)
           recuperadaEnergia2=[recuperadaEnergia2(1:end-dif) zeros(1,dif)];
           senialEnergia2=[senialEnergia2 zeros(1,dif)];
        else
           senialEnergia2=[senialEnergia2(1:end-dif) zeros(1,dif)];
           recuperadaEnergia2=[recuperadaEnergia2 zeros(1,dif)];
        end
        % ahora si, se hace la correlacion con las seniales del mismo tamanio
        correlacion2=corr(senialEnergia2',recuperadaEnergia2');

        % se normalizan las seniales
        recuperadaEnergia2=normalizacion(recuperadaEnergia2')';
        senialEnergia2=normalizacion(senialEnergia2')';
        if(numImpul2==6)
           numCoeficientes=length(coeficientes2);

coeficientes_energia=energyCoeficient(senialEnergia2,impulsos2,numImpul2,longDep);
        end
          % se llama a la interfaz para mostrar los resultados obtenidos
        PFCanalisisWavelet(senialEnergia2,recuperadaEnergia2,
numImpul2,correlacion2,coeficientes2,longDep,nivel);
     % en caso de que la seÃ±al venga en energia, se devuelven directamente
     % los resultados
     else
        recuperada=normalizacion(recuperada')';
        senialEnergia=normalizacion(senialEnergia')';
        PFCanalisisWavelet(senialEnergia,recuperada,
numImpul,correlacion,coeficientes,longDep,nivel);
     end
   end
```

analisisDaubechies(senialEnergia,longDep,nivelTope,energia,tipoM)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Author: José María Lucas Zaragoza
% Date: 22 Augost 2010
% Este metodo es el encargado de hacer el analisis haciendo uso de las
% funciones wavelet de matlab, el tipo de wavelet seleccionado debe ser
% Coiflets
% Parametros de entrada:
%   - senialEnergia: senial de entrada a la cual realizar el anÃ¡lisis
%   - longDep: longitud de la seÃ±al depurada
%   - nivelTope: nivel maximo a realizar el analisis
%   - energia: booleano(0 o 1) que indica si la seÃ±al de entrada es en
%     energia o en tiempo
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function senial=analisisDaubechies(senialEnergia,longDep,nivelTope,energia,tipoM)
%tipo de analisis
tipo=['db' tipoM];
% se realiza la transformada para cada uno de los niveles
for nivel=1:nivelTope
    % se construyen los parametros para el filtrado
    cadenaNivel=['.' int2str(nivel)];
    wname = [tipo cadenaNivel];
    % se hallan los coeficientes del filtrada
    [r] = dbwavf(wname);
    [LO_D,HI_D,LO_R,HI_R] = orthfilt(r);
    % se obtienen los coeficientes de deconstrucion y construccion
    coeficientes= filter(LO_D,1,senialEnergia);
    % y se obtienen las seÃ±ales tras el filtrado
    recuperada=filter(LO_R,1,coeficientes);
    coeficientesD= filter(HI_D,1,senialEnergia);
    recuperadaD=filter(HI_R,1,coeficientesD);

     recuperada=normalizacion(recuperada')';
    % se usan los coeficientes de aproximacion para la reconstruccion
    [impulsos,numImpul]=deteccionImpulsosEnergia(recuperada,longDep);
    % se obtienen la correlacion entre la senial recuperada e inicial
    correlacion=corr(senialEnergia',recuperada');
    %si la seÃ±al no esta en energia, lo pasamos a energia para comparar
    %resultados
    if(~energia)
        % No debemos cambiar el nÃºmero de muestras de la seÃ±al de energÃa
        % cada vez,debemos seleccionar el nÃºmero de muestras al inicio
        muestras=625;
        save muestras.mat muestras
```

```
        % se detecta el numero de muestras optimo para detectar los 6
        % impulsos
        detmuestras(senialEnergia);
        % se depura la seÃ±al recuperada en aproximacion anterior (se toma
        % la energia de la senial
        [recuperadaEnergia2,senialDepRecuperada]=depurar(recuperada');
        [senialEnergia2,senialDepEnergia]=depurar(senialEnergia');
        % longitud de la senial depurada
        longDep=length(senialDepRecuperada);
        % se detectan los impulsos de la senial
        [impulsos2,numImpul2]=deteccionImpulsosEnergia(recuperadaEnergia2,longDep);


        coeficientes2=r;
        % puede suceder que la senial inicial y la depurada no tengan la
        % misma longitud (tienen el mismo numero de muestras, pero entre el
        % primer y el ultimo impulsos puede que varie un poco.
        dif=length(senialEnergia2)-length(recuperadaEnergia2);
        if(dif<0)
           recuperadaEnergia2=[recuperadaEnergia2(1:end-dif) zeros(1,dif)];
           senialEnergia2=[senialEnergia2 zeros(1,dif)];
        else
           senialEnergia2=[senialEnergia2(1:end-dif) zeros(1,dif)];
           recuperadaEnergia2=[recuperadaEnergia2 zeros(1,dif)];
        end

        correlacion2=corr(senialEnergia2',recuperadaEnergia2');
        % se normalizan las seniales
        recuperadaEnergia2=normalizacion(recuperadaEnergia2')';
        senialEnergia2=normalizacion(senialEnergia2')';
        if(numImpul2==6)
           numCoeficientes=length(coeficientes)*2;

coeficientes_energia=energyCoeficient(senialEnergia2,impulsos2,numImpul2,longDep);
        end
        % se llama a la interfaz para mostrar los resultados obtenidos
        PFCanalisisWavelet(senialEnergia2,recuperadaEnergia2,
numImpul2,correlacion2,coeficientes2,longDep,nivel);
    % en caso de que la seÃ±al venga en energia, se devuelven directamente
    % los resultados
    else
        if(numImpul==6)
           numCoeficientes=length(coeficientes)*2;
           coeficientes_energia=energyCoeficient(senialEnergia,impulsos,numImpul,longDep);
        end

PFCanalisisWavelet(senialEnergia,recuperada,numImpul,correlacion,coeficientes,longDep,ni
vel);
    end
    %aqui si la seÃ±al no es en energia
end
```

creaImpulsos(localizacion,valor,longitud)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Author: José MarÃa Lucas Zaragoza
% Date: 22 Augost 2010
% Este metodo se encarga de dibujar los impulsos que se encuentrar en la
% señal que se esta analizando
% Parametros de entrada:
%   - localizacion: situacion de los impulsos dentro de la senial analizada
%   - valor: vector con la energia de los impulsos
%   - longitud: longitud de la senial
% Parametros de salida:
%   - Senial con los impulsos dibujados
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function vectorSalida=creaImpulsos(localizacion,valor,longitud)
% se crea vector auxiliar
vectorAux=zeros(1,longitud);
% se recorren la posicion de los impulsos para crear el nuevo vector
for pulsoInicio=1:length(localizacion(1,:))

    vectorAux(localizacion(1,pulsoInicio):localizacion(2,pulsoInicio))=valor(pulsoInicio);
end
% se devuelve el vector con los impulsos
vectorSalida=vectorAux;
```

depurar(senial)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Author: José María Lucas Zaragoza
% Date: 22 Augost 2010
% Esta función se va a encargar de eliminar las muestras de la senial que
% no contienen información, limitandose esta a las muestras que se
% encuentran entre el primer y el Ãºltimo impulso.
% Parametros de entrada:
%   - senial: senial de entrada en tiempo
% Parametros de salida:
%   - senialEnergia: senial depurada final en energia
%   - senialDep: senial depurada en tiempo
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
function [senialEnergia,senialDep]=depurar(senial)

global longEfectiva;
% se cargan las muestras que debemos tomar de la seÃ±al para detectar los 6
% impulsos
load muestras.mat
% se hace uso del metodo de deteccion de impulsos de la seÃ±al en tiempo
% para detectar el inicio y el final de la seÃ±al depurada
[inicio,fin,senialEnergia]=detInicioFin_New(senial,0,muestras);
% se devuelve la seÃ±al depurada en tiempo
senialDep=senial(inicio:fin);
longEfectiva=length(senialEnergia);
%se salva la senial en energia
save senialEnergia.mat senialEnergia
```

deteccionImpulsosEnergia(senialEnergia,lon_se)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Author: José María Lucas Zaragoza
% Date: 22 Augost 2010
% Este mÃ©todo se encarga, de detectar los impulsos de la seÃ±al de entrada
% (debe ser una seÃ±al en energÃa)
% Parametros de entrada:
%   - senialEnergia: senial en energÃa de la que se deben extraer los impulsos
%   - lon_se: numero de muestras de la senial en energia
% Parametros de salida:
%   - impulsos: impulsos detectados
%   - numImpulsos: numero de impulsos detectados
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [impulsos,numImpulsos]=deteccionImpulsosEnergia(senialEnergia,lon_se)

%umbral de detecciÃ³n
um = 0.003;
%senial en energia
en=senialEnergia;
%longitud de la seÃ±al en energÃa (que contiene los 6impulsos)
global longEfectiva;

%nivel de umbral para seleccion
pto=max(en)*um;
%binarizacion de la seÃ±al la energia que sea mayor a pto se hace
```

```
%1 lo que no se hace 0, teniendo la seÃ±al con niveles logicos se
%hace una evaluacion de Ã±a misma para saber cuando empiezan y
%cuando terminan los impulsos de la seÃ±al de vibraciÃ³n
en=en>pto;
j=1;
k=1;
sgtoi(j)=1;
sgtof(k)=1;
dif_t(j)=1;
%sabemos que el primer impulso va a estar en la primera muestra, debido a
%la forma de hacer la depuracion de la seÃ±al
 if(en(1)==1)
     sgtoi(j)=1;
     j=j+1;
 end

%para el resto de las muestras hacemos la deteccion (tambien se sabe que el
%ultimo impulsos debe estar en la ultima muestra
for i=2:length(en)
   %encuentro inicio
      if en(i-1)<en(i)
      sgtoi(j)=i;
      j=j+1;
   %encuentro final
      elseif en(i-1)>en(i)
      sgtof(k)=i;
      k=k+1;
      end
end


%ya se tiene donde empiezan y terminan los impulsos
if length (sgtoi)~=length (sgtof)
   %Evalua si los segmetos de las energias son
   %del mismo tamaÃ±o de lo contrario existe
   %un error y se corrige
   disp 'correciÃ³n en la deteccciÃ³n de los impulsos'

   if(length(sgtof)<length(sgtoi))
      sgtof=[sgtof longEfectiva];
   else
      sgtoi=[1 sgtoi];
   end
end
% se organizan los impulsos
impulsos(1,:)=sgtoi;
impulsos(2,:)=sgtof;
% se toma el numero de impulsos
numImpulsos=(length(sgtoi));
```

detInicioFin_New(varargin)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Author: JosÃ© MarÃa Lucas Zaragoza
% Date: 22 Augost 2010
% Este mÃ©todo se encarga, de detectar los impulsos de la seÃ±al de entrada
% (debe ser una seÃ±al en tiempo), se usa para la depuraciÃ³n, detectando el
% pulso de inicio y de fin.
% Parametros de entrada:
%   - varargin(1): senial en tiempo
%   - varargin(2): habilitar grÃ¡fica para la energia de la seÃ±al
%   - varargin(3): muestras de la seÃ±al de entrada
% Parametros de salida:
%   - inicio: donde empieza el primer impulsos
%   - fin: donde termina el ultimo pulso
%   - senialEnergia: calculo de la energia de la senial de entrada
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [inicio,fin,senialEnergia]=detInicioFin_New(varargin)
% frecuencia de muestreo
fs = 50000;
% umbral
um = 0.003;
% se evaluan el numero de parametros que se toman a la entrada
if nargin < 3, tr_ev =1/624;else tr_ev=1/varargin{3}; end
% se toman los parametros
senal=varargin{1};
lon_se=length(senal);
enable_gr=varargin{2};

%Se calcula el primer tiempo final del calculo de la energia
%y se guarda la variable en tf(1) vector de tiempos finales(en muestras)
tf(1)=lon_se*tr_ev;
%ti es el vector en el que se guarda los tiempos iniciales de los calculos
%de energia (en muestras)
ti(1)=1;
%normalizado de la senial
senal=normalizacion(senal);
%se calcula la energia del primer tramo
en(1)=energia(senal(round(ti(1)):round(tf(1))),fs,1);%primer calculo de energia
n=1;
%El siguiente while calcula la energia de la seÃ±al de forma en tramos
```

```
%establecidos por tr_ev.

% hace los respectivos calculos hasta que la variable n supere la longitud
% de la seÃ±al
while (tf(n)<lon_se)
   n=n+1;          %Incrementa n
   ti(n)=tf(n-1);   %El valor del tiempo final de la seleccion pasa a ti
   tf(n)=tf(n-1)+lon_se*tr_ev; %Se recalcula el valor de tf para el tramo
   if tf(n)>lon_se
      en(n)=energia(senal(round(ti(n)):end),fs,1);
   else
   %Llamado de funciÃ³n que calcula la energia del tramo
   en(n)=energia(senal(round(ti(n)):round(tf(n))),fs,1);

   end
end

factor=max(senal)/max(en); %calculamos la amplitud en energia respecto la amplitud de
vibracion
en2=en.*factor;          %ES UNA NORMALIZACION EN ENERGIA RESPECTO SEÃ‘AL
                         %Se hace para la grafica superpuesta llevando la
                         %amplitud de la seÃ±al de energia a la amplitud
                         %de la seÃ±al de vibraciÃ³n (bueno esto...)
tvida1=linspace(0,lon_se/fs,length(en));% tvida es un vector con la informacion
                         %de la seÃ±al de energia representada en tiempo
                         %y no en numero de muestras

if enable_gr==1          %Evalua si la grafica de la energia es pedida por
                         %el usuario
   set(gcf,'Color','W')   %pone en color blanco la imagen para las graficas
   tvida=linspace(0,lon_se/fs,lon_se);%Vector con la informacion en tiempo
                         %para la correcta representacion en tiempo de la
                         %seÃ±al de vibraciÃ³n
   plot(tvida,senal,tvida1,en2,'r');
   title('SeÃ±al de vibracion')
   ylabel('AceleraciÃ³n en m/s^2')
   xlabel('Tiempo en segundos')
   grid on
end

senialEnergia=en;

% a continuaciÃ³n el proceso de deteccion

pto=max(en)*um;%nivel de umbral para seleccion


en=en>pto;%binarizacion de la seÃ±al la energia que sea mayor a pto se hace
        %1 lo que no se hace 0, teniendo la seÃ±al con niveles logicos se
        %hace una evaluacion de Ã±a misma para saber cuando empiezan y
        %cuando terminan los impulsos de la seÃ±al de vibraciÃ³n
```

```
j=1;
k=1;
sgtoi(j)=1;
sgtof(k)=1;
dif_t(j)=1;
for i=2:n
   %encuentro inicio
   if en(i-1)<en(i)
      sgtoi(j)=i;
      j=j+1;
   %encuentro final
   elseif en(i-1)>en(i)
      sgtof(k)=i;
      k=k+1;
   end
end
%ya se tiene donde empiezan y terminan los impulsos
if length (sgtoi)~=length (sgtof)%Evalua si los segmetos de las energias son
                        %del mismo tamaÃ±o de lo contrario existe
                        %un error y se corrige
   disp 'correcciÃ³n en la detecciÃ³n de impulsos'
   sgtof=sgtof(2:length (sgtof));
end

%inicio y fin de la seÃ±al depurada
inicio=round((sgtoi(1)-1)*lon_se*tr_ev);
fin=round(sgtof(end)*lon_se*tr_ev);
%se devuelve la seÃ±al en energia
senialEnergia=senialEnergia(sgtoi(1):sgtof(end));
```

detmuestras(varargin)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Author: José MarÃía Lucas Zaragoza
% Date: 22 Augost 2010
% Este metodo se encarga de seleccion del numero optimo de muestras que se
% debe tomar de la señal de entrada, para detectar 6 impulsos. Es un metodo
% iterativo y por ello exige los siguientes parametros.
% Parametros de entrada:
%   - varargin(1): senial de entrada (en tiempo)
%   - varargin(2): 1 para imprimir la senial y 0 para que no sea asi
%   - varargin(3): muestras detectadas hasta ahora
% Parametros de salida(se ha usado :
%   - inicio: inicio de la senial detectada hasta el momento
%   - fin: fin de la senial detectada hasta el momento
%   - senialEnergia: senialEnergia con las muestras que se han seleccionado
%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
function [inicio,fin,senialEnergia,muestras]=detmuestras(varargin)

% parametros para el anÃ¡lisis
fs = 50000;
um = 0.003;
% si no se define ningÃºn numero de muestras a la entrada, se pone un valor
% por defecto
if nargin < 3, tr_ev =1/624;else tr_ev=varargin{3}; end
% se define como global la variable muestras, ya que se va cambiando
global muestras;
muestras=1/tr_ev;
senal=varargin{1};
lon_se=length(senal);
% iteraciones es usado para detectar el primer "hilo" que sale del metodo
% (para almacenar las muestras que deben tomarse)
global iteraciones;
iteraciones=1;
senal=varargin{1};
enable_gr=0;
% este metodo es analogo a la detInicioFin, solo que va variando las
% muestras en la deteccion.

% Se calcula el primer tiempo final del calculo de la energia
% y se guarda la variable en tf(1) vector de tiempos finales(en muestras)
tf(1)=lon_se*tr_ev;

% ti es el vector en el que se guarda los tiempos iniciales de los calculos
% de energia (en muestras)
ti(1)=1;
senal=normalizacion(senal);

en(1)=energia(senal(round(ti(1)):round(tf(1))),fs,1);%primer calculo de energia
n=1;
% El siguiente while calcula la energia de la seÃ±al de forma en tramos
%establecidos por tr_ev.

% hace los respectivos calculos hasta que la variable n
% supere la longitud de la seÃ±al
while (tf(n)<lon_se)
   % Incrementa n
   n=n+1;
   % El valor del tiempo final de la seleccion pasa a ti
   ti(n)=tf(n-1);
```

```
    % Se recalcula el valor de tf para el tramo
    tf(n)=tf(n-1)+lon_se*tr_ev;
    if tf(n)>lon_se
       en(n)=energia(senal(round(ti(n)):end),fs,1);
    else
    % Llamado de funciÃ³n que calcula
    % la energia del tramo
    en(n)=energia(senal(round(ti(n)):round(tf(n))),fs,1);
    end
end
% calculamos la amplitud en energia respecto la amplitud de vibracion
factor=max(senal)/max(en);

% Se hace para la grafica superpuesta llevando la
% amplitud de la seÃ±al de energia a la amplitud
% de la seÃ±al de vibraciÃ³n

en2=en.*factor;

% tvida es un vector con la informacion
% de la seÃ±al de energia representada en tiempo
% y no en numero de muestras

tvida1=linspace(0,lon_se/fs,length(en));
% Evalua si la grafica de la energia es pedida por
% el usuario
if enable_gr==1
    % pone en color blanco la imagen para las graficas
    set(gcf,'Color','W')
    % Vector con la informacion en tiempo
    % para la correcta representacion en tiempo de la
    % seÃ±al de vibraciÃ³n
    tvida=linspace(0,lon_se/fs,lon_se);
    plot(tvida,senal,tvida1,en2,'r');
    title('SeÃ±al de vibracion')
    ylabel('AceleraciÃ³n en m/s^2')
    xlabel('Tiempo en segundos')
    grid on
end
senialEnergia=en;


pto=max(en)*um;% nivel de umbral para seleccion

% binarizacion de la seÃ±al la energia que sea mayor a pto se hace
% 1 lo que no se hace 0, teniendo la seÃ±al con niveles logicos se
% hace una evaluacion de Ã±a misma para saber cuando empiezan y
% cuando terminan los impulsos de la seÃ±al de vibraciÃ³n
en=en>pto;

j=1;
k=1;
```

```
sgtoi(j)=1;
sgtof(k)=1;
dif_t(j)=1;
for i=2:n
   % encuentro inicio
   if en(i-1)<en(i)
      sgtoi(j)=i;
      j=j+1;
   % encuentro final
   elseif en(i-1)>en(i)
      sgtof(k)=i;
      k=k+1;
   end
end
% ya tenemos donde empiezan y terminan los impulsos
if length (sgtoi)~=length (sgtof)
  % Evalua si los segmetos de las energias son
  % del mismo tamaÃ±o de lo contrario existe
  % un error y se corrige
  disp 'correcciÃ³n en la detecciÃ³n de impulsos'
  sgtof=sgtof(2:length (sgtof));
end
% aqui se tiene realmente los impulsos que hay
lsf=length(sgtof);

% inicio y fin de los impulsos
inicio=round((sgtoi(1)-1));
fin=round(sgtof(end));

%en caso de no detectarse los seis impulsos se umenta el numero de muestras
%y se hace una llamada iterativa
if(lsf<6)
   muestras=muestras+5;
   %en caso de superar las 900 muestras se aumenta mas rÃ¡pido el num. de
   %muestras
   if(muestras>900)
      muestras=muestras+95;
   end
   tr_ev=1/muestras;
   detmuestras(senal,0,tr_ev);

end

% Si es la Ãºltima vez que se modifican las muestras, se salvan
if(iteraciones==1)
   save muestras.mat muestras
else
   return
end
iteraciones=iteraciones+1;
```

energia(varargin)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Author: José María Lucas Zaragoza
% Date: 22 Augost 2010
% Este método se encarga, de calcular la energia de la senial (o tramo que
% se le pasa)
% Parametros de entrada:
%   - varargin(1): senial de entrada
%   - varargin(2): frecuencia de muestreo de la senial
%   - opc(3): ( 1=en frecuencia 2=en tiempo) tipo de calculo de energia de
%   la senial
% Parametros de salida:
%   - Energia de la senial
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


function [varargout]=energia(varargin)

% Toma de los parametros de entrada
senal=varargin{1};
N=length(senal);
% debemos poner frecuencia de muestreo solo si el anÃ¡lisis es en frecuencia
fs=varargin{2};
opc=varargin{3};
%si pasan la seÃ±al vacia entones la energia es puesta a 0
if N==0
   En=0;
else
   % para calculo de la energia en frecuencia
   if opc==2
      En=sum(abs(senal).^2)*(fs/N);
   % para calculo de la energia en tiempo
   elseif opc==1
      En=sum((senal).^2)/N;
   end
end
%se devuelve la energia de la senial
```

```
varargout{1}=En;
```

energyCoeficient(SenialActual,impulsos,numImpul,lon_se)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Author: José María Lucas Zaragoza
% Date: 22 Augost 2010
% Este mÃ©todo se encarga, de calcular la energia de los impulsos
% Parametros de entrada:
%   - SenialActual: senial en energia analizada
%   - impulsos: impulsos detectados (localizacion dentro de la seÃ±al)
%   - numImpul: numero de impulsos detectados
%   - lon_se: longitud de la seÃ±al depurada en tiempo
% Parametros de salida:
%   - coeficientes_energia: vector con la energia de los impulsos
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [coeficientes_energia]=energyCoeficient(SenialActual,impulsos,numImpul,lon_se);
% se comprueba si es la primera deteccion correcta de los impulsos, ya que
% pueden ser que en otro nivel tambien se hallan detectado

load energiaCoeficientes.mat;
condicion=strcmp(energiaCoeficientes,'No se han detectado los 6 impulsos necesarios');

% Se detecta la energia de los impulsos y el tiempo entre ellos
for i=1:numImpul
    senialAux=SenialActual(impulsos(1,i):impulsos(2,i));
    coeficientes_energia(i)=energia(senialAux,500000,1);
    if(i<6)

tiempoEntreImpulsos(i)=length(impulsos(2,i):impulsos(1,i+1))*lon_se/640*(1/500000);
    end
end
% Se imprime por pantalla
disp('En energyCoeficient')
disp(['Los coeficientes en energia son: ' num2str(coeficientes_energia)]);

%en caso de que no se hallan detectado los seis impulsos anteriormete
%cogemos lo hallado en el paso anterior
if(condicion)
```

```
    energiaCoeficientes=(coeficientes_energia);
    tiempoImpulsos=(tiempoEntreImpulsos);
    %si es la segunda vez que se detectan (otro nivel), se aÃ±aden
else
    energiaCoeficientes=[energiaCoeficientes;coeficientes_energia];
    tiempoImpulsos=[tiempoImpulsos;tiempoEntreImpulsos];
end
% salvamos la energia y el tiempo para posterior uso
save energiaCoeficientes;
save tiempoImpulsos;

% se imprime la distancia entre impulsos
disp(['La distancia entre impulsos es: ' num2str(tiempoEntreImpulsos)]);
```

MiInterfaz

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Author: José María Lucas Zaragoza
% Date: 22 Augost 2010
% Este metodo froma parte de la presentacion del PFC
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function MiInterfaz
% se elimina todo lo que se ha realizado anteriormente
clear,clc,cla,close all
% se carga el fondo
[fondo,map]=imread('fondoPresentacion.jpg','jpg');
% se toma el tamanio
[alto,ancho,nuse]=size(fondo);

% Aqui se crea la ilustracióndonde se representa luego

figdiag=figure('Units','Pixels','Position',[alto,ancho,1.5*alto,1*ancho],...
'Number','off','Name','Presentecion PFC','Menubar','none',...
'color',[0 0 0]);

% Se ubican los ejes en figura
axes('Units','Normalized',...
'Position',[0 0 1 1]);
% se centra
centerfig(figdiag,0)

% se coloca
image(fondo),colormap(map),axis off,hold on
```

```
% se pone el texto de la presentacion sobre la imagen
text(70,50,'Estudio de la seÃ±al de vibraciÃ³n (seÃ±al OLTC)
','Fontname','Arial','Fontsize',15,'Fontangle','Italic', ...
'Fontweight','Bold','color',[0 0 1]);
text(130,70,' con la transformada de wavelet
','Fontname','Arial','Fontsize',15,'Fontangle','Italic', ...
'Fontweight','Bold','color',[0 0 1]);

% nombre del autor
ext(115,350,'Por: José María Lucas Zaragoza','Fontname', ...
'Comic Sans MS','Fontangle','Italic','Fontweight','Bold', ...
'Fontsize',14,'color',[0 0 0]);

%  creaciÃ³n botÃ³n Continuar
normalizacionX=380/800;
normalizacionY=170/1280;

% se crea a donde debe dirigir el boton, cuando es pulsado
botok=uicontrol('Style','pushbutton', ...
'Units','normalized', ...
'Position',[.27 .15 normalizacionX normalizacionY], ...
'String','',...
'Callback','clear all; close all;clc; PFCanalisisWavelet;');

% imagen sobre el boton
[a,map]=imread('ejecucion.jpg');
[r,c,d]=size(a);
x=ceil(r/66);
y=ceil(c/266);
g=a(1:x:end,1:y:end,:);
g(g==255)=5.5*255;
set(botok,'CData',g);
```

normalizacion(ref1)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Author: JosÃ© MarÃa Lucas Zaragoza
% Date: 22 Augost 2010
% Este metodo se encarga de la normalizaciÃ³n de la seÃ±al
% Parámetro de entrada:
%     -ref1: señal de entrada
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function[ref1_norm]=normalizacion(ref1)

maximo=max(ref1);
ref1_norm=ref1/maximo;
```

PFCanalisisWavelet(varargin)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Author: José María Lucas Zaragoza
% Date: 22 Augost 2010
% Metodo encargado de las operaciones que tienen lugar en la interdaz
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function varargout = PFCanalisisWavelet(varargin)
% PFCANALISISWAVELET M-file for PFCanalisisWavelet.fig
%      PFCANALISISWAVELET, by itself, creates a new PFCANALISISWAVELET or
raises the existing
%      singleton*.
%
%      H = PFCANALISISWAVELET returns the handle to a new
PFCANALISISWAVELET or the handle to
%      the existingsesingleton*.
%
%      PFCANALISISWAVELET('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in PFCANALISISWAVELET.M with the given input
arguments.
%
%      PFCANALISISWAVELET('Property','Value',...) creates a new
PFCANALISISWAVELET or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before PFCanalisisWavelet_OpeningFunction gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to PFCanalisisWavelet_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Copyright 2002-2003 The MathWorks, Inc.
```

```
% Edit the above text to modify the response to help PFCanalisisWavelet

% Last Modified by GUIDE v2.5 26-Sep-2010 19:56:59

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
   'gui_Singleton',  gui_Singleton, ...
   'gui_OpeningFcn', @PFCanalisisWavelet_OpeningFcn, ...
   'gui_OutputFcn',  @PFCanalisisWavelet_OutputFcn, ...
   'gui_LayoutFcn',  [] , ...
   'gui_Callback',   []);
if nargin && ischar(varargin{1})
   gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
   [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
   gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before PFCanalisisWavelet is made visible.
% Tipo 1 de parÃ¡metros de entrada: (Cuando ya se ha seleccionado la seÃ±al)
%   senialOriginal;senialRecuperada;impulsos;correlacion;niveles;lon_se1;impulsos;
%   energyCoef;datosOptimizados;posicionAumento;longEfectiva.
% Tipo 2 de parÃ¡metros de entrada: Cuando no hay seÃ±al seleccionada
%   no existen parÃ¡metros, tan solo se coloca las imagenes y botones para
%   dar el aspecto de wavemenu a la interfaz.

function PFCanalisisWavelet_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to PFCanalisisWavelet (see VARARGIN)

global posicionGrafsenial;


%Se comprueba en primer lugar, si se ejecuta la funcion por primera vez
%En caso de que se tenga el anÃ¡lisis realizado
if(get(handles.checkAna,'Value'))
   %oculto la representaciÃ³n de la seÃ±al anterior
   posicionGrafSenial=get(handles.grafSenial,'Position');
   set(handles.grafSenial,'Position',[-50 -50 1 1]);
   %Cojo los parametros de entrada
   global senialOriginal;
   global senialRecuperada;
   global impulsos;
```

```
global correlacion;
global niveles;
global lon_se1;
global impulsos;
global energyCoef;
global datosOptimizados;
global posicionAumento;
global longEfectiva;
senialOriginal=cell2mat(varargin(1));
senialOriginal=senialOriginal(1:longEfectiva);
senialRecuperada=cell2mat(varargin(2));
senialRecuperada=senialRecuperada(1:longEfectiva);
numImpulsos=cell2mat(varargin(3));
correlacion=cell2mat(varargin(4));
coeficientes=cell2mat(varargin(5));
lon_se1=cell2mat(varargin(6));
niveles=cell2mat(varargin(7));

%en caso de que se tengan 5 niveles
if(niveles==5)
   %se muestra la seÃ±al en la grÃ¡fica donde estaba el nivel 1
   niveles=1;
   set(handles.impulsosn1,'String','impulsos n5:');
   set(handles.corr1,'String','Correlation n5:' );
   %en caso contrario se hace el anÃ¡lisis normal
else
   set(handles.impulsosn1,'String','impulsos n1:');
   set(handles.corr1,'String','Correlation n1:' );
end

%en caso de seleccionarse el optimizado, se representan solo las
%seÃ±ales donde hay 6 impulsos detectados.
 if(get(handles.optimize,'Value'))
   %miro si estÃ¡n todos los impulsos, si no, no se pintan
   if(numImpulsos==6)
      impulsos=cell2mat(varargin(8));
      energyCoef=cell2mat(varargin(9));
      %se pintan los impulsos
      vectorImpulsos=creaImpulsos(impulsos,energyCoef,length(senialRecuperada));
      datosOptimizados=1;
   end
 else
      %si no se tienen 6impulsos, no se pinta la seÃ±al
      energyCoef=0;
      impulsos=0;
      datosOptimizados=0;
end

%Obtenidos los parÃ¡metros se recolocan las seÃ±ales
if(get(handles.recolocar,'Value'))
   load posiciones;
end
```

```
%Se coloca cada uno de los niveles en una posiciÃ³n distinta
switch(niveles)
   %Nivel 1
   case{1}
      if(get(handles.recolocar,'Value'))
         set(handles.graf1,'Position',posiciones(1,:));
         posiciones(1,:);
      end
      set(handles.graf1,'Visible','on')
      axes(handles.graf1)
      cla
      set(handles.aumento1,'Visible','on')
      set(handles.impulsos1,'String',numImpulsos);
      set(handles.correlacion1,'String',correlacion);
      coeficientes1=coeficientes;
      clear senialNivel1
      senialNivel1=senialRecuperada;
      set(handles.aumento1,'Position',posicionAumento(1,:));

      save senialNivel1.mat senialNivel1;

      save coef1.mat coeficientes1
      %Nivel 2
   case{2}
      if(get(handles.recolocar,'Value'))
         set(handles.graf2,'Position',posiciones(2,:));
         posiciones(2,:);
      end
      set(handles.graf2,'Visible','on')
      axes(handles.graf2)
      cla
      set(handles.aumento2,'Visible','on')
      set(handles.impulsos2,'String',numImpulsos);
      set(handles.correlacion2,'String',correlacion);
      coeficientes2=coeficientes;
      clear senialNivel2
      senialNivel2=senialRecuperada;
      save senialNivel2.mat senialNivel2;
      save coef2.mat coeficientes2
      set(handles.aumento2,'Position',posicionAumento(2,:));
      %Nivel 3
   case{3}
      if(get(handles.recolocar,'Value'))
         set(handles.graf3,'Position',posiciones(3,:));
      end
      axes(handles.graf3)
      cla
      set(handles.graf3,'Visible','on')
      set(handles.aumento3,'Visible','on')
      set(handles.impulsos3,'String',numImpulsos);
      set(handles.correlacion3,'String',correlacion);
      coeficientes3=coeficientes;
```

```
        clear senialNivel3
        senialNivel3=senialRecuperada;
        save senialNivel3.mat senialNivel3;
        save coef3.mat coeficientes3
        set(handles.aumento3,'Position',posicionAumento(3,:));
        %Nivel 4
    case{4}
        if(get(handles.recolocar,'Value'))
            set(handles.graf4,'Position',posiciones(4,:));
        end
        axes(handles.graf4)
        cla
        set(handles.graf4,'Visible','on')
        set(handles.aumento4,'Visible','on')
        set(handles.impulsos4,'String',numImpulsos);
        set(handles.correlacion4,'String',correlacion);
        coeficientes4=coeficientes;
        clear senialNivel4
        senialNivel4=senialRecuperada;
        save senialNivel4.mat senialNivel4;
        save coef4.mat coeficientes4
        set(handles.aumento4,'Position',posicionAumento(4,:));
end
%Una vez colocados los ejes se representan
%Por defecto el salvado de los coeficientes estarÃ¡ habilitado
set(handles.save,'Enable','on')
set(handles.saveCoeficientes,'Enable','on')
%frecuencia de muestreo
fs=50000;
%ejes para representar
global ejeX;
ejeX=linspace(0,lon_se1/fs,length(senialOriginal));
ejeX2=linspace(0,lon_se1/fs,length(senialRecuperada));

plot(ejeX,senialOriginal,'b');
hold on;
plot(ejeX2,senialRecuperada,'r');

%en caso de que se estÃ© optimizando, se representan los impulsos
if(datosOptimizados)
    hold on;
    plot(ejeX2,vectorImpulsos,'g')
end
grid on;
whitebg('white');
legend('Senial inicial energia','Senial recuperada','location','NorthWest')
whitebg('black');
color('white');

%El siguiente switch permite el avance adecuado de la barra de proceso
switch niveles
    case 1
```

```
            set(handles.evolucion,'String','level 1 plotted')
            %debo ver el tipo de anÃ¡lisis para coger el nivel correcto
            habilitado=get(handles.tipoMatlab,'Enable');
            if(strcmp(habilitado,'off'))
               porcentage=100*niveles/get(handles.nivel,'Value');
            else
               porcentage=100*niveles/get(handles.nivelAux,'Value');
            end
            set(handles.progreso,'Value',porcentage);
         case 2
            set(handles.evolucion,'String','level 2 plotted')
            habilitado=get(handles.tipoMatlab,'Enable');
            %debo ver el tipo de anÃ¡lisis para coger el nivel correcto
            if(strcmp(habilitado,'off'))
               porcentage=100*niveles/get(handles.nivel,'Value');
            else
               porcentage=100*niveles/get(handles.nivelAux,'Value');
            end

            set(handles.progreso,'Value',porcentage);
         case 3
            set(handles.evolucion,'String','level 3 plotted')
            %debo ver el tipo de anÃ¡lisis para coger el nivel correcto
            habilitado=get(handles.tipoMatlab,'Enable');
            if(strcmp(habilitado,'off'))
               porcentage=100*niveles/get(handles.nivel,'Value');
            else
               porcentage=100*niveles/get(handles.nivelAux,'Value');
            end

            set(handles.progreso,'Value',porcentage);
         case 4
            set(handles.evolucion,'String','level 4 plotted')
            %debo ver el tipo de anÃ¡lisis para coger el nivel correcto
            habilitado=get(handles.tipoMatlab,'Enable');
            if(strcmp(habilitado,'off'))
               porcentage=100*niveles/get(handles.nivel,'Value');
            else
               porcentage=100*niveles/get(handles.nivelAux,'Value');
            end

            set(handles.progreso,'Value',porcentage);
      end
   % Una vez analizado el caso de que la seÃ±al este analizada, en este otro se
   % tiene que permitir la optenciÃ³n de la seÃ±al, y posteriormente llamar al
   % mÃ©todo que realizarÃ¡ todos los calculos para la realizaciÃ³n del anÃ¡lisis,
   % esta vez, ya si, pasara a la parte anterior para mostrar los datos.
   else
      % lo primero que se hace es colocar todas la imÃ¡genes que hacen que la
      % interfaz tenga un aspecto parecido a wavemenu
```

```
    % Choose default command line output for PFCanalisisWavelet
    handles.output = hObject;

    % Update handles structure
    guidata(hObject, handles);

    axes(handles.panelLateral)
    panelLat=imread('BarraDerecha2.bmp');
    axis off;
    imshow(panelLat);
    hold on

    axes(handles.representaciones)
    represen=imread('Centro.bmp');
    axis off;
    imshow(represen);
    hold on



    %Se pone la interfaz en el medio de la pantalla
    scrsz=get(0,'ScreenSize');
    pos_act=get(gcf,'Position');
    xr=scrsz(3)-pos_act(3);
    xp=round(xr/2);
    yr=scrsz(4)-pos_act(4);
    yp=round(yr/2);
    set(gcf,'Position', [49 7 pos_act(3) pos_act(4)]);


    %Se coloca en cada botÃ³n una imagen

    [aumen,map]=imread('aumentar.jpg');
    [r,c,d]=size(aumen);
    x=ceil(r/30);
    y=ceil(c/30);
    g=aumen(1:x:end,1:y:end,:);
    g(g==255)=5.5*255;
    set(handles.aumento1,'CData',g);
    set(handles.aumento2,'CData',g);
    set(handles.aumento3,'CData',g);
    set(handles.aumento4,'CData',g);

end

% Fin del mÃ©todo principal, los siguientes son los manejadores que hacen
% capaz la interactuaciÃ³n con la interfaz y sus respectivas operaciones

% --- Outputs from this function are returned to the command line.
function varargout = PFCanalisisWavelet_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes during object creation, after setting all properties.
function figure1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to figure1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called




% --- Executes on button press in file.
function file_Callback(hObject, eventdata, handles)
% hObject    handle to file (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


%BotÃ³n para introducir la seÃ±al de entrada



% La siguientes funciÃ³n se encarga del tipo de transformada cuando se hace
% el anÃ¡lisis matricial

% --- Executes on selection change in tipoTrans.
function tipoTrans_Callback(hObject, eventdata, handles)
% hObject    handle to tipoTrans (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%si se ha seleccionado un tipo matricial se prepara la interfaz para tomar
%los niveles y las pestaÃ±as adecuadas
if (get(handles.tipoTrans,'Value')<5)
   set(handles.nivelAux,'Visible','off')
   set(handles.nivel,'Enable','on')
   set(handles.nivel,'Visible','on')

   set(handles.tipoMatlab,'Enable','off');
   set(handles.optimize,'Value',0);
   set(handles.optimize,'Visible','on');
   set(handles.energiaOff,'Visible','off');
%en caso contrario se prepara para las funciones matlab
else
   set(handles.nivelAux,'Visible','on')
   set(handles.nivel,'Enable','on')
   set(handles.nivel,'Visible','off')
```

```
    set(handles.nivel,'Value',1)

    set(handles.optimize,'Value',0);
    set(handles.tipoMatlab,'Enable','on');
    set(handles.optimize,'Visible','off');
    set(handles.energiaOff,'Visible','on');
end

% --- Executes during object creation, after setting all properties.
function tipoTrans_CreateFcn(hObject, eventdata, handles)
% hObject    handle to tipoTrans (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


% --- Executes on selection change in nivel.
function nivel_Callback(hObject, eventdata, handles)
% hObject    handle to nivel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns nivel contents as cell array
%        contents{get(hObject,'Value')} returns selected item from nivel


% --- Executes during object creation, after setting all properties.
function nivel_CreateFcn(hObject, eventdata, handles)
% hObject    handle to nivel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end

% Esta funciÃ³n se ejecuta cuando pulsamos el botÃ³n para realizar el
% anÃ¡lisis

% --- Executes on button press in analize.
function analize_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to analize (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% se ponen los valores por defecto a los valores que se muestran en el reporte
% para informar de que no se ha realizado el anÃ¡lisis
vacio='';
save vacio;
load vacio;
coef1='No se ha realizado el anÃ¡lisis de dichos coeficientes';
coef2='No se ha realizado el anÃ¡lisis de dichos coeficientes';
coef3='No se ha realizado el anÃ¡lisis de dichos coeficientes';
coef4='No se ha realizado el anÃ¡lisis de dichos coeficientes';
energiaCoeficientes='No se han detectado los 6 impulsos necesarios';
tiempoImpulsos='No se han detectado los 6 impulsos necesarios';
save tiempoImpulsos.mat;
save energiaCoeficientes.mat;
save coef1;
save coef2;
save coef3;
save coef4;

% habilitamos los valores a mostrar tras el analisis
set(handles.checkAna,'Value',1);
set(handles.correlation,'Enable','on');
set(handles.impulsos,'Enable','on');
set(handles.report,'Enable','on');

global longDep
global senialEnergia
global transformada
global posicionAumento;


% si no estamos analizando con las funciones de matlab se coge el nivel del
% desplegable principal
if(get(handles.tipoTrans,'Value')<5)
   nivel=get(handles.nivel,'Value');
else
   nivel=get(handles.nivelAux,'Value');
end

% cada vez que hay un anÃ¡lisis se reinicializan todos los valores


set(handles.correlacion1,'String','---');
set(handles.impulsos1,'String','---');

axes(handles.graf1)
cla

set(handles.correlacion2,'String','---');
```

```
set(handles.impulsos2,'String','---');

axes(handles.graf2)
cla

set(handles.correlacion3,'String','---');
set(handles.impulsos3,'String','---');


axes(handles.graf3)
cla

set(handles.correlacion4,'String','---');
set(handles.impulsos4,'String','---');

axes(handles.graf4)
cla
%tipo de transformada
transformada=get(handles.tipoTrans,'Value');
%en el caso de que seleccione la opcion tipo 5 es por que es de tipo MATLAB
%entonces meto un booleano que indique que ademas habrÃ¡ otro tipo de
%seleccion, que es el tipo de funcion matlab a tratar

if(transformada==5)
   tipoFmatlab(1)=get(handles.tipoMatlab,'Value');
   tipoFmatlab(2)=get(handles.energiaOff,'Value');
   save tipoFmatlab.mat tipoFmatlab;
end
optimizar=get(handles.optimize,'Value');
%una vez tenemos todos los parametros de entrada se llama a la funcion
%encargada de realizar el anÃ¡lisis. Es decir, la interdaz ya ha realizado
%la la labor de obtener los parÃ¡metros de entrada
pruebaConInterfaz(longDep,senialEnergia,nivel,transformada,optimizar);




% --- Executes on button press in checkAna.
function checkAna_Callback(hObject, eventdata, handles)
% hObject    handle to checkAna (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of checkAna


%La funciÃ³n siguiente es la encargada del aumento de la grÃ¡fica 1

% --- Executes on button press in aumento1.
function aumento1_Callback(hObject, eventdata, handles)
% hObject    handle to aumento1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)
%%Debo esconder TODAS las grÃ¡ficas y representar en grafsenial

global posicionAumento;
%se comprueba si hay alguna seÃ±al aumentada
if(~get(handles.aumentada,'Value'))

   global posicionGrafSenial;

   set(handles.graf2,'Position',[-50 -50 1 1]);
   set(handles.graf3,'Position',[-50 -50 1 1]);
   set(handles.graf4,'Position',[-50 -50 1 1]);
   set(handles.aumento2,'Position',[-50 -50 1 1]);
   set(handles.aumento3,'Position',[-50 -50 1 1]);
   set(handles.aumento4,'Position',[-50 -50 1 1]);
   set(handles.aumento1,'Position',[108.2 37.7 2.8 1.154])
   %pintamos en los ejes grandes (encima en realidad)
   set(handles.graf1,'Position',posicionGrafSenial);
   set(handles.aumentada,'Value',1);

else

   load posiciones
   %si estaba aumentada tengo que hacer la opentaciÃ³n contraria
   set(handles.graf2,'Position',posiciones(2,:));
   set(handles.graf3,'Position',posiciones(3,:));
   set(handles.graf4,'Position',posiciones(4,:));
   set(handles.graf1,'Position',posiciones(1,:));
   set(handles.aumento1,'Position',posicionAumento(1,:));
   set(handles.aumento2,'Position',posicionAumento(2,:));
   set(handles.aumento3,'Position',posicionAumento(3,:));
   set(handles.aumento4,'Position',posicionAumento(4,:));
   set(handles.aumentada,'Value',0);

end

%La funciÃ³n siguiente es la encargada del aumento de la grÃ¡fica 2

% --- Executes on button press in aumento2.
function aumento2_Callback(hObject, eventdata, handles)
% hObject    handle to aumento2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global posicionAumento;
if(~get(handles.aumentada,'Value'))

   posicion2=get(handles.aumento2,'Position');
   global posicionGrafSenial
   set(handles.graf1,'Position',[-50 -50 1 1]);
   set(handles.graf3,'Position',[-50 -50 1 1]);
   set(handles.graf4,'Position',[-50 -50 1 1]);
   set(handles.aumento1,'Position',[-50 -50 1 1]);
```

```
    set(handles.aumento3,'Position',[-50 -50 1 1]);
    set(handles.aumento4,'Position',[-50 -50 1 1]);
    set(handles.aumento2,'Position',[108.2 37.7 2.8 1.154]);
    %pintamos en los ejes grandes (encima en realidad)
    set(handles.graf2,'Position',posicionGrafSenial);
    set(handles.aumentada,'Value',1);
else
    load posiciones
    %si estaba aumentada tengo que hacer la opentaciÃ³n contraria
    set(handles.graf2,'Position',posiciones(2,:));
    set(handles.graf3,'Position',posiciones(3,:));
    set(handles.graf4,'Position',posiciones(4,:));
    set(handles.graf1,'Position',posiciones(1,:));
    set(handles.aumento1,'Position',posicionAumento(1,:));
    set(handles.aumento2,'Position',posicionAumento(2,:));
    set(handles.aumento3,'Position',posicionAumento(3,:));
    set(handles.aumento4,'Position',posicionAumento(4,:));
    set(handles.aumentada,'Value',0);
end

%La funciÃ³n siguiente es la encargada del aumento de la grÃ¡fica 3

% --- Executes on button press in aumento3.
function aumento3_Callback(hObject, eventdata, handles)
% hObject    handle to aumento3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global posicionAumento;
if(~get(handles.aumentada,'Value'))

    global posicionGrafSenial
    set(handles.graf2,'Position',[-50 -50 1 1]);
    set(handles.graf1,'Position',[-50 -50 1 1]);
    set(handles.graf4,'Position',[-50 -50 1 1]);
    set(handles.aumento3,'Position',[108.2 37.7 2.8 1.154]);
    set(handles.aumento1,'Position',[-50 -50 1 1]);
    set(handles.aumento2,'Position',[-50 -50 1 1]);
    set(handles.aumento4,'Position',[-50 -50 1 1]);
    %pintamos en los ejes grandes (encima en realidad)
    set(handles.graf3,'Position',posicionGrafSenial);
    set(handles.aumentada,'Value',1);
else

    load posiciones
    %si estaba aumentada tengo que hacer la opentaciÃ³n contraria
    set(handles.graf2,'Position',posiciones(2,:));
    set(handles.graf3,'Position',posiciones(3,:));
    set(handles.graf4,'Position',posiciones(4,:));
    set(handles.graf1,'Position',posiciones(1,:));
    set(handles.aumento1,'Position',posicionAumento(1,:));
    set(handles.aumento2,'Position',posicionAumento(2,:));
    set(handles.aumento3,'Position',posicionAumento(3,:));
```

```
    set(handles.aumento4,'Position',posicionAumento(4,:));
    set(handles.aumentada,'Value',0);
end

%La funciÃ³n siguiente es la encargada del aumento de la grÃ¡fica 4

% --- Executes on button press in aumento4.
function aumento4_Callback(hObject, eventdata, handles)
% hObject    handle to aumento4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global posicionAumento;
if(~get(handles.aumentada,'Value'))

    global posicionGrafSenial
    set(handles.graf2,'Position',[-50 -50 1 1]);
    set(handles.graf3,'Position',[-50 -50 1 1]);
    set(handles.graf1,'Position',[-50 -50 1 1]);
    set(handles.aumento4,'Position',[108.2 37.7 2.8 1.154]);
    set(handles.aumento1,'Position',[-50 -50 1 1]);
    set(handles.aumento3,'Position',[-50 -50 1 1]);
    set(handles.aumento2,'Position',[-50 -50 1 1]);
    %pintamos en los ejes grandes (encima en realidad)
    set(handles.graf4,'Position',posicionGrafSenial);
    set(handles.aumentada,'Value',1);
else
    load posiciones
    %si estaba aumentada tengo que hacer la opentaciÃ³n contraria
    set(handles.graf2,'Position',posiciones(2,:));
    set(handles.graf3,'Position',posiciones(3,:));
    set(handles.graf4,'Position',posiciones(4,:));
    set(handles.graf1,'Position',posiciones(1,:));
    set(handles.aumento1,'Position',posicionAumento(1,:));
    set(handles.aumento2,'Position',posicionAumento(2,:));
    set(handles.aumento3,'Position',posicionAumento(3,:));
    set(handles.aumento4,'Position',posicionAumento(4,:));
    set(handles.aumentada,'Value',0);
end



% --- Executes on button press in recolocar.
function recolocar_Callback(hObject, eventdata, handles)
% hObject    handle to recolocar (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of recolocar
```

```
% --- Executes on button press in pushbutton10.
function pushbutton10_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%El siguiente boton se encarga de hacer visibles las correlaciones

% --- Executes on button press in correlation.
function correlation_Callback(hObject, eventdata, handles)
% hObject    handle to correlation (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%si pulso hago visibles las correlaciones

if(~strcmp(get(handles.correlacion1,'String'),'Not evaluated'))
   set(handles.corr1,'Visible','on')
   set(handles.correlacion1,'Visible','on')
end
if(~strcmp(get(handles.correlacion2,'String'),'Not evaluated'))
   set(handles.corr2,'Visible','on')
   set(handles.correlacion2,'Visible','on')
end
if(~strcmp(get(handles.correlacion3,'String'),'Not evaluated'))
   set(handles.corr3,'Visible','on')
   set(handles.correlacion3,'Visible','on')
end
if(~strcmp(get(handles.correlacion4,'String'),'Not evaluated'))
   set(handles.corr4,'Visible','on')
   set(handles.correlacion4,'Visible','on')
end


% --- Executes during object creation, after setting all properties.
function graf1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to graf1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: place code in OpeningFcn to populate graf1


%La siguiente funciÃ³n hace visibles las cadenas con el nÃºmero de impulsos

% --- Executes on button press in impulsos.
function impulsos_Callback(hObject, eventdata, handles)
% hObject    handle to impulsos (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
if(~strcmp(get(handles.impulsos1,'String'),'Not evaluated'))
   set(handles.impulsosn1,'Visible','on')
```

```
    set(handles.impulsos1,'Visible','on')
end
if(~strcmp(get(handles.impulsos2,'String'),'Not evaluated'))
    set(handles.impulsosn2,'Visible','on')
    set(handles.impulsos2,'Visible','on')
end
if(~strcmp(get(handles.impulsos3,'String'),'Not evaluated'))
    set(handles.impulsosn3,'Visible','on')
    set(handles.impulsos3,'Visible','on')
end
if(~strcmp(get(handles.impulsos4,'String'),'Not evaluated'))
    set(handles.impulsosn4,'Visible','on')
    set(handles.impulsos4,'Visible','on')
end


% A continuaciÃ³n tenemos las funciones necesarias para completar la barra
% con los menus


% --------------------------------------------------------------------
function FileDeMenu_Callback(hObject, eventdata, handles)
% hObject    handle to FileDeMenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%--------------------------------------------------------------------

%
% %Esto se mete en el menu de lectura de fichero
% function entradaSenial2_Callback(hObject, eventdata, handles)
activo=get(handles.salvadoCoef,'checked');
if(strcmp(activo,'on'))

    set(handles.save,'Enable','on')
else

    set(handles.save,'Enable','off')
end
if(get(handles.tipoTrans,'Value')==5)
    set(handles.save,'Enable','off');
end


% --------------------------------------------------------------------
function prueba1_Callback(hObject, eventdata, handles)
% hObject    handle to prueba1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --------------------------------------------------------------------
```

```
function medidaDirecta_Callback(hObject, eventdata, handles)
% hObject    handle to medidaDirecta (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --------------------------------------------------------------------
function entradaSenial_Callback(hObject, eventdata, handles)
% hObject    handle to entradaSenial (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



% --------------------------------------------------------------------
function entradaSenial2_Callback(hObject, eventdata, handles)
% hObject    handle to entradaSenial2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



% --------------------------------------------------------------------
function help_Callback(hObject, eventdata, handles)
% hObject    handle to help (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



% --------------------------------------------------------------------
function senial_Callback(hObject, eventdata, handles)
% hObject    handle to senial (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



% --------------------------------------------------------------------
function saveCoeficientes_Callback(hObject, eventdata, handles)
% hObject    handle to saveCoeficientes (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



% --------------------------------------------------------------------
function save_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to save (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.pdf,'Enable','off')

% La siguiente funciÃ³n es la encargada de tratar la seÃ±al de entrada

% --------------------------------------------------------------------
function signal_Callback(hObject, eventdata, handles)
% hObject    handle to signal (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% menÃº para seleccionar la seÃ±al
[FileName Path]=uigetfile({'*.mat'},'Seleccionar una seÃ±al de entrada');
if isequal(FileName,0)
   %en caso de que no se halla seleccionado ninguna seÃ±al
   errordlg({'No ha seleccionado ninguna seÃ±al'},'Mensaje de error')
   % se vuelve a deshabilitar todos los componentes que implicar el
   % anÃ¡lisis de una seÃ±al que no se ha introducido
   set(handles.dataSize,'String','SeÃ±al Entrada');
   set(handles.tipoTrans,'Enable','off')
   set(handles.nivel,'Enable','off')
   set(handles.optimize,'Enable','off')
   set(handles.analize,'Enable','off')
   set(handles.report,'Enable','off')
   set(handles.optimize,'Value',0);
   return
   % en caso de que si que halla una seÃ±al introducida
else
   % se deshabilitan las opciones de guardado
   set(handles.save,'Enable','off')
   set(handles.saveCoeficientes,'Enable','off')
   set(handles.optimize,'Value',0);
   %se lee la seÃ±al
   ruta=strcat(Path,FileName);
   senialAux1=load(ruta);
   senialAux2=struct2array(senialAux1);
   % senialAux2=cell2mat(struct2cell([senialAux1]));
   [fila,columna]=size(senialAux2);
   global senial;
   % se comprueba si la seÃ±al es tal cual se mide o si ya estÃ¡ tratada
   if(fila>1)
      if(columna>4)
         senial=senialAux2(:,5);
      else
         if(columna==1)
            senial=senialAux2(:,1);
            %en cualquier otro caso se muestra un mensaje de error
         else
            errordlg({'No ha seleccionado una seÃ±al valida, para mas informaciÃ³n pulse
"help"'},'Mensaje de error')
            return
```

```
            end
        end
    else
        % si tiene un nÃºmero pequeÃ±o de muestras tambiÃ©n se muestra el
        % mensaje de error
        if(columna<100)
            errordlg({'No ha seleccionado una seÃ±al valida, para mas informaciÃ³n pulse
"help"'},'Mensaje de error')
            return
        else
        senial=senialAux2(1,:);
        senial=senial';
        end
    end
    % se toma la seÃ±al de entrada
    senialAnalizada=FileName;
    % se salva para su uso posterior
    save senialAnalizada.mat senialAnalizada

    %debo coger las posiciones iniciales al principio
    if(~get(handles.checkAna,'Value'))
        posicionGraf1=get(handles.graf1,'Position');
        posicionGraf2=get(handles.graf2,'Position');
        posicionGraf3=get(handles.graf3,'Position');
        posicionGraf4=get(handles.graf4,'Position');
        %se salvan para cuando se necesite recolocarlas
        posiciones=[posicionGraf1;posicionGraf2;posicionGraf3;posicionGraf4];
        save posiciones.mat posiciones;
        %se guardar las posiciones de los botones
        global posicionAumento;

        posicionAumento(1,:)=get(handles.aumento1,'Position');
        posicionAumento(2,:)=get(handles.aumento2,'Position');
        posicionAumento(3,:)=get(handles.aumento3,'Position');
        posicionAumento(4,:)=get(handles.aumento4,'Position');
    end
    %compruebo si no accedo el botÃ³n tras un primer anÃ¡lisis
    if(get(handles.checkAna,'Value'))

        % si es asÃ se deben restablecer todas las condiciones iniciales de la
        % interfaz, una vez se comprueba que he introducido una seÃ±al valida

        % Se deben volver a mover todas las grÃ¡ficas

        set(handles.graf1,'Position',[-50 -50 1 1]);
        set(handles.graf2,'Position',[-50 -50 1 1]);
        set(handles.graf3,'Position',[-50 -50 1 1]);
        set(handles.graf4,'Position',[-50 -50 1 1]);

        %se deben recolocar las graficas para la posterior representacion
        set(handles.recolocar,'Value',1)
```

```
    %tambien debo de hacer invisible las correlaciones (si estÃ¡n visibles)
    %y los botones de aumentar grÃ¡fica
    set(handles.aumento1,'Visible','off')
    set(handles.aumento2,'Visible','off')
    set(handles.aumento3,'Visible','off')
    set(handles.aumento4,'Visible','off')
    set(handles.corr1,'Visible','off')
    set(handles.corr2,'Visible','off')
    set(handles.corr3,'Visible','off')
    set(handles.corr4,'Visible','off')
    set(handles.correlacion1,'Visible','off')
    set(handles.correlacion2,'Visible','off')
    set(handles.correlacion3,'Visible','off')
    set(handles.correlacion4,'Visible','off')
    set(handles.impulsos1,'Visible','off')
    set(handles.impulsosn1,'Visible','off')
    set(handles.impulsos2,'Visible','off')
    set(handles.impulsosn2,'Visible','off')
    set(handles.impulsos3,'Visible','off')
    set(handles.impulsosn3,'Visible','off')
    set(handles.impulsos4,'Visible','off')
    set(handles.impulsosn4,'Visible','off')
    set(handles.aumento1,'Visible','off')
    set(handles.aumento2,'Visible','off')
    set(handles.aumento3,'Visible','off')
    set(handles.aumento4,'Visible','off')
    %Y ahora se ponen a not evaluated
    set(handles.correlacion1,'String','Not evaluated')
    set(handles.correlacion2,'String','Not evaluated')
    set(handles.correlacion3,'String','Not evaluated')
    set(handles.correlacion4,'String','Not evaluated')
    set(handles.impulsos1,'String','Not evaluated')
    set(handles.impulsos2,'String','Not evaluated')
    set(handles.impulsos3,'String','Not evaluated')
    set(handles.impulsos4,'String','Not evaluated')
end
    set(handles.analize,'Enable','on');

    global pAux;
    global senialEnergia
    global senialDep
    global muestras;

    muestras=625;
    save muestras.mat muestras
    % esta funciÃ³n se encarga de ver las muestras necesarias para la
    % detecciÃ³n de los 6 impulsos
    detmuestras(senial);
    % primero se depura la seÃ±al ya que se pasarÃ¡ la seÃ±al en energÃa
    [senialEnergia,senialDep]=depurar(senial);
    % si ha habido una detecciÃ³n de un nÃºmero de muestras diferencte a
    % 624 se carga este nÃºmero
```

```
        pAux=fopen('muestras.mat');
        if(pAux>0)
           load muestras.mat
        else
           muestras=624;
        end
        % se pone en la interfaz la seÃ±al usada y su nÃºmero de muestras
        cadenaAux=[FileName ' (' int2str(muestras) ')'];
        set(handles.dataSize,'String',cadenaAux);

        %voy a comprobar que las muestras seleccionadas no son mas de 624*2
        %en caso de que sea asÃ deshabilito las funciones matriciales, ya
        %que no tendrÃ¡n sentido al ser muy pesadas.
        pAux=fopen('muestras.mat');
        if(pAux>0)
           load muestras.mat
           if(muestras>1248)
              % deshabilito los mÃ©todos matriciales y habilito las
              % funciones matlab
              set(handles.tipoTrans,'Enable','off')
              set(handles.tipoTrans,'Value',5)
              set(handles.tipoMatlab,'Enable','on')
              % se tiene que habilitar el nivel de las funciones de matlab
              set(handles.nivelAux,'Visible','on')
              set(handles.nivel,'Visible','off')
              % tambien se pone en la pestaÃ±a de proceso que han sido
              % cambiados el nÃºmero de muestras
              set(handles.evolucion,'String',['se han tomado mas muestras en la seÃ±al de entrada
para una correcta detecciÃ³n --> ' int2str(muestras)]);
           else
              set(handles.nivel,'Visible','on')
              set(handles.nivelAux,'Visible','off')
              set(handles.nivel,'Enable','on')
              set(handles.tipoTrans,'Value',1)
              set(handles.tipoTrans,'Enable','on')
              set(handles.tipoMatlab,'Enable','off')
           end
        end
        %esta es la frecuencias de muestreo de la seÃ±al
        fs = 50000;
        lon_se=length(senialDep);
        lon_se2=length(senialEnergia);

        %se calculan los impulsos de la seÃ±al de entrada
        [impulsos,numImpul]=deteccionImpulsosEnergia(senialEnergia,lon_se2);

        set(handles.impulsos,'Enable','on');
        set(handles.impulsos1,'String',numImpul);
        set(handles.correlation,'Enable','off');
        set(handles.report,'Enable','off');
        factor=max(senialDep)/max(senialEnergia);
        % se calcula la amplitud en energia respecto la amplitud de
```

```
        % vibracion
        senialEnergia=senialEnergia.*factor;
        x1=linspace(0,lon_se/fs,lon_se);
        x2=linspace(0,lon_se/fs,lon_se2);
        % se guarda la senial en energia para pasarla luego como parÃ¡metro
        global longDep
        longDep=length(senialDep);

        % cuando se tiene la seÃ±al se pregunta si se quiere representar
        opc=questdlg('Â¿Desea representar la seÃ±al?','Representacion','Si','No','Si');
        % aqui se hace la representaciÃ³n de la seÃ±al
    if strcmp(opc,'No')

    else
        % como se desea representar, se tiene que recolocar la grÃ¡fica,
        % chequeo condicion
        global posicionGrafSenial

        if(get(handles.checkAna,'Value'))
            set(handles.grafSenial,'Position',posicionGrafSenial);
        else
            posicionGrafSenial=get(handles.grafSenial,'Position');
        end



        % Se eligen los ejes
        axes(handles.grafSenial);
        cla

        %aqui se representan conjuntamente la seÃ±al en energia y
        %depurada

        whitebg('black')
        plot(x1,senialDep,'r')
        hold on
        plot(x2,senialEnergia,'b')
        grid on
        xlabel('tiempo(s)');
        ylabel('Amp. seÃ±al');
        whitebg('white');
        legend('Senial inicial depurada','Senial en energia','location','NorthWest')
        whitebg('black');
        color('white');
        set(handles.progreso,'Value',10)
        set(handles.evolucion,'String','Signal in plotted, waiting for signal analysis')
    end
end

% ahora se habilitan los botones de seleccciÃ³n de transformada y nivel
set(handles.nivel,'Enable','on');
set(handles.analize,'Enable','on');
```

```
set(handles.optimize,'Enable','on');



% --- Executes on button press in aumentada.
function aumentada_Callback(hObject, eventdata, handles)
% hObject    handle to aumentada (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of aumentada

% La siguiente funciÃ³n se ejecuta la pulsar el boton report, el cual se
% encarga de la generaciÃ³n del documento con las caracteristicas del
% anÃ¡lisis

% --- Executes on button press in report.
function report_Callback(hObject, eventdata, handles)
% hObject    handle to report (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
clc
global senial;
global senialDep;
global x1;
global senialOriginal;
global senialEnergia;
global x2;
global lon_se1;

% Los parÃ¡metros necesarios para mostrar son:
%1. Transformada y mÃ©todo
%2. NÃºmero de niveles
%3. Correlaciones
%4. Impulsos

global senialRecuperada;
global numImpulsos;
global correlacion;
global impulsos;
global energyCoef;
global datosOptimizados;

tipoAnalisis=get(handles.tipoTrans,'Value');

metodo='Uso de operaciones matriciales';
niveles=get(handles.nivel,'Value');
if(tipoAnalisis==5)
      tipoAnalisis=get(handles.tipoMatlab,'Value');
```

```
        metodo='Funciones Matlab';
        niveles=get(handles.nivelAux,'Value');
end

switch tipoAnalisis
   case 1
      tipoAnalisis='haar';
   case 2
      tipoAnalisis='db2';
   case 3
      tipoAnalisis='db3';
   case 4
      tipoAnalisis='coiflets';
   case 5
      tipoAnalisis='biorthogonal';
end

% se toman los valores a imprimir
correlaciones=zeros(1,niveles);
numImpulsos=zeros(1,niveles);

for i=1:niveles
   switch i
      case 1
         correlaciones(i)=str2double(get(handles.correlacion1,'String'));
         numImpulsos(i)=str2num(get(handles.impulsos1,'String'));
      case 2
         correlaciones(i)=str2double(get(handles.correlacion2,'String'));
         numImpulsos(i)=str2num(get(handles.impulsos2,'String'));
      case 3
         correlaciones(i)=str2double(get(handles.correlacion3,'String'));
         numImpulsos(i)=str2num(get(handles.impulsos3,'String'));
      case 4
         correlaciones(i)=str2double(get(handles.correlacion4,'String'));
         numImpulsos(i)=str2num(get(handles.impulsos4,'String'));
   end
end
nivelesTotales=1:1:niveles;
% se salvan para despues mostrarlas
save tabla.mat metodo tipoAnalisis nivelesTotales numImpulsos correlaciones
load tabla
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% se toma el archivo
niv=get(handles.nivel,'Value');
dirActual=pwd;
archivo=[dirActual '\prueba0.html'];
archivoDoc=[dirActual '\prueba0.rtf'];
% cojo este fichero por que si existe algÃºn fichero anterior, este estarÃ¡
fichero=fopen('prueba0.rtf');
% se pregunta si se quieren borran los demas archivos creados
```

```
if(fichero~=-1)
   opc=questdlg(['Â¿ Desea borrar los reportes anteriores ?' '                    (no--
>genera uno nuevo)']);

   %Aqui se hace la representaciÃ³n de la seÃ±al
   if (strcmp(opc,'No')|(strcmp(opc,'Cancel')))

   else
      warning off
      fclose('all');
      delete([dirActual '\*.rtf']);
      delete([dirActual '\*.doc']);
      delete([dirActual '\*.XML']);
      warning on
   end
end
% se genera el reporte
if(datosOptimizados)

fichero=pintaReporte(senialDep,x1,senialEnergia,x2,niv,lon_se1,senialOriginal,datosOptimiz
ados,impulsos,energyCoef);
else

fichero=pintaReporte(senialDep,x1,senialEnergia,x2,niv,lon_se1,senialOriginal,datosOptimiz
ados);
end
% una vez generado el reporte se pone 100% a la barra de progreso y se
% actualiza el estado
set(handles.progreso,'Value',100);
set(handles.evolucion,'String',' Report generated ')
ficheroNuevo=[dirActual 'pruebaActual.doc'];
% se crea un fichero nuevo para mostrar el reporte actual
[SUCCESS,MESSAGE,MESSAGEID]=copyfile(fichero,ficheroNuevo,'f');
% para mostrar donde ha sido creado, preguntar si se quiere mostrar y en
% caso afirmativo abrirlo
opc=questdlg(['El reporte ha sido creado en:                    '...
   ficheroNuevo ' ' '                                          Â¿Desea abrirlo?']);
%Aqui se hace la representaciÃ³n de la seÃ±al
if (strcmp(opc,'No')|(strcmp(opc,'Cancel')))
   return;
else
   open(ficheroNuevo);
end


% --- Executes on slider movement.
function progreso_Callback(hObject, eventdata, handles)
% hObject    handle to progreso (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
```

```
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider


% --- Executes during object creation, after setting all properties.
function progreso_CreateFcn(hObject, eventdata, handles)
% hObject    handle to progreso (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background, change
%       'usewhitebg' to 0 to use default.  See ISPC and COMPUTER.
usewhitebg = 1;
if usewhitebg
   set(hObject,'BackgroundColor',[.9 .9 .9]);
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


% --- Executes during object creation, after setting all properties.
function grafSenial_CreateFcn(hObject, eventdata, handles)
% hObject    handle to grafSenial (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: place code in OpeningFcn to populate grafSenial


% --- Executes during object creation, after setting all properties.
function graf3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to graf3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: place code in OpeningFcn to populate graf3


% --- Executes during object creation, after setting all properties.
function graf4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to graf4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: place code in OpeningFcn to populate graf4


% --- Executes during object creation, after setting all properties.
function graf2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to graf2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
% Hint: place code in OpeningFcn to populate graf2


% --------------------------------------------------------------------
function salvadoCoef_Callback(hObject, eventdata, handles)
% hObject    handle to salvadoCoef (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)Ã§
if(strcmp(get(handles.salvadoCoef,'checked'),'on'))
   set(handles.salvadoCoef,'checked','off')
else
   set(handles.salvadoCoef,'checked','on')
end

% --------------------------------------------------------------------
function options_Callback(hObject, eventdata, handles)
% hObject    handle to options (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% PestaÃ±a de optimizaciÃ³n del mÃ©todo matricial

% --- Executes on button press in optimize.
function optimize_Callback(hObject, eventdata, handles)
% hObject    handle to optimize (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of optimize

%si le doy a optimizar debo deshabilitar el nivel que queremos

if(get(handles.optimize,'Value'))
   %lo ponemos a 4 para analizar todos los niveles
   set(handles.nivel,'Value',4)
   set(handles.nivel,'Enable','off')
else
   set(handles.nivel,'Value',1)
   set(handles.nivel,'Enable','on')
end


% --- Executes on selection change in tipoMatlab.
function tipoMatlab_Callback(hObject, eventdata, handles)
% hObject    handle to tipoMatlab (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns tipoMatlab contents as cell array
%        contents{get(hObject,'Value')} returns selected item from tipoMatlab
```

```
% --- Executes during object creation, after setting all properties.
function tipoMatlab_CreateFcn(hObject, eventdata, handles)
% hObject    handle to tipoMatlab (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


% PestaÃ±a de energiaOff en el anÃ¡lisis de funciones matlab

% --- Executes on button press in energiaOff.
function energiaOff_Callback(hObject, eventdata, handles)

% si se ha seleccionado, que no se haga el anÃ¡lisis de la seÃ±al en energÃa
if(~get(handles.energiaOff,'Value'))

    %se pone al valor actual en caso de que se halla manipulado anteriormente
    nivelEnAux=get(handles.nivelAux,'Value');
    if(nivelEnAux==5)
       nivelEnAux=4;
    end
    set(handles.nivel,'Value',nivelEnAux)
    set(handles.nivel,'Visible','on')
    set(handles.nivel,'Enable','on')

else
    set(handles.nivel,'Visible','off')
    set(handles.nivelAux,'Visible','on')

end


% --- Executes on selection change in nivelAux.
function nivelAux_Callback(hObject, eventdata, handles)
% hObject    handle to nivelAux (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns nivelAux contents as cell array
%        contents{get(hObject,'Value')} returns selected item from nivelAux
valorNivel=get(handles.nivelAux,'Value');
set(handles.nivel,'Value',valorNivel)
```

```
% --- Executes during object creation, after setting all properties.
function nivelAux_CreateFcn(hObject, eventdata, handles)
% hObject    handle to nivelAux (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


% El siguiente mÃ©todo se encarga de crear el reporte con los coeficientes
% analizados

% --------------------------------------------------------------------
function doc_Callback(hObject, eventdata, handles)
% hObject    handle to doc (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

load coef1.mat;
load coef2.mat;
load coef3.mat;
load coef4.mat;

report coeficientes
dirActual=pwd;
archivoCreado=[dirActual '\coeficientes.doc'];
copyFile('coeficientes.rtf','coeficientes.doc','f')

opc=questdlg(['El reporte ha sido creado en: '...
   'coeficientes.doc' ' ' 'Â¿Desea abrirlo?']);
if (strcmp(opc,'No')|(strcmp(opc,'Cancel')))

   return;
else
   open(archivoCreado);
end

% En esta funcion se habrÃa creado el reporte en pdf

% --------------------------------------------------------------------
function pdf_Callback(hObject, eventdata, handles)
% hObject    handle to pdf (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
load coef1.mat;
load coef2.mat;
load coef3.mat;
load coef4.mat;

report coeficientes
copyFile('coeficientes.rtf','coeficientes.pdf','f')
dirActual=pwd;
archivoCreado=[dirActual '\coeficientes.pdf'];
opc=questdlg(['El reporte ha sido creado en: '...
   'coeficientes.pdf' ' ' 'Â¿Desea abrirlo?']);
if (strcmp(opc,'No')|(strcmp(opc,'Cancel')))

   return;
else
   open(archivoCreado);
end


% --------------------------------------------------------------------
function Ayuda_Callback(hObject, eventdata, handles)
% hObject    handle to Ayuda (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)




% --------------------------------------------------------------------
function Help_Callback(hObject, eventdata, handles)
% hObject    handle to Help (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

open('manual.doc')
```

pintaReporte(varargin)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Author: JosÃ© MarÃa Lucas Zaragoza
% Date: 22 Augost 2010
% Este metodo es usado para la redaccion y ajuste del documento reportado
% Parametros de entrada:
```

```
%   - varargin(1): senial depurada
%   - varargin(2): eje senial depurada
%   - varargin(3): senial en energia
%   - varargin(4): eje senial en energia
%   - varargin(5): nivel maximo analizado
%   - varargin(6): longitud de la senial
%   - varargin(7): senial original
%   - varargin(8): boolenano si los datos estan optimizados
% Parametros de salida:
%   - fichero: normbre del fichero donde esta localizado el reporte
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function fichero=pintaReporte(varargin)

% se muestra la pantalla de comandos para que se muestre el proceso de
% generaciÃ³n del reporte
commandwindow
% cambio de formato para la idonea generaciÃ³n del reporte
format long
% toma de los parametros de entrada
senialDep=cell2mat(varargin(1));
ejeSenialDep=cell2mat(varargin(2));
senialEnergia=cell2mat(varargin(3));
ejeSenialEne=cell2mat(varargin(4));
nivel=cell2mat(varargin(5));
lonse=cell2mat(varargin(6));
senialOriginal=cell2mat(varargin(7));
datosOptimizados=cell2mat(varargin(8));
% parametros globales que han variado
global ImpulsosDetectados;
global Energia;
% en caso de que los datos esten optimizados (se detecten los 6 impulsos)
if(datosOptimizados)
    % se preparan las sentencias a mostrar en la salida
    impulsos=cell2mat(varargin(9));
    energia=cell2mat(varargin(10));
    ImpulsosDetectados1=[ 'Los impulsos de la seÃ±al son: ' '(1.) De ' int2str(impulsos(1,1)) ' a
' int2str(impulsos(2,1))];
    Energia1=[ 'Poseen una energia de: (1.) ' num2str(energia(1))];
    ImpulsosDetectados2=[ ' ' '(2.) De ' int2str(impulsos(1,2)) ' a ' int2str(impulsos(2,2))];
    Energia2=[ ' (2.) ' num2str(energia(2))];
    ImpulsosDetectados3=[ ' ' '(3.) De ' int2str(impulsos(1,3)) ' a ' int2str(impulsos(2,3))];
    Energia3=[ ' (3.) ' num2str(energia(3))];
    ImpulsosDetectados4=[ ' ' '(4.) De ' int2str(impulsos(1,4)) ' a ' int2str(impulsos(2,4))];
    Energia4=[' (4.) ' num2str(energia(4))];
    ImpulsosDetectados5=[ ' ' '(5.) De ' int2str(impulsos(1,5)) ' a ' int2str(impulsos(2,5))];
    Energia5=[ ' (5.) ' num2str(energia(5))];
    ImpulsosDetectados6=[ ' ' '(6.) De ' int2str(impulsos(1,6)) ' a ' int2str(impulsos(2,6))];
    Energia6=[ ' (6.) ' num2str(energia(6))];
```

```
   ImpulsosDetectados=[ImpulsosDetectados1 ImpulsosDetectados2  ImpulsosDetectados3
ImpulsosDetectados4  ImpulsosDetectados5  ImpulsosDetectados6]
   Energia=[Energia1 Energia2 Energia3 Energia4 Energia5 Energia6];
end
fs=50000;

% se caragan las senial pertinentes
switch nivel
   case 1
      load senialNivel1.mat
   case 2
      load senialNivel1.mat
      load senialNivel2.mat
   case 3
      load senialNivel1.mat
      load senialNivel2.mat
      load senialNivel3.mat
   case 4
      load senialNivel1.mat
      load senialNivel2.mat
      load senialNivel3.mat
      load senialNivel4.mat
end

% se cargan las seniales en energia
energia=fopen('energiaCoeficientes.mat');
if(energia>-1)
load energiaCoeficientes.mat;
end
% y el tiempo entre impulsos
timpulsos=fopen('tiempoImpulsos.mat');
if(timpulsos>-1)
load tiempoImpulsos.mat;
end

load tabla.mat;
% se genera el reporte
report prueba.rpt;
fichero=ans;
```

pruebaConinterfaz(longDep,senialEntrada,nivel,eleccionIn,optimizar)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Author: Jose Maria Lucas Zaragoza
% Date: 22 Augost 2010
```

```
% Metodo encargado de realizar el análisis y devolución de resultados a la
% interfaz
% ParÃ¡metros de entrada...
% longDep: longitud de la seÃ±al depurada
% senialEntrada: senial de entrada a analizar
% nivel: nivel mÃ¡ximo a realizar el anÃ¡lisis
% eleccionIn: tipo de anÃ¡lisis
% optimizar: si se desea optimizar o no
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


function out=pruebaConInterfaz(longDep,senialEntrada,nivel,eleccionIn,optimizar)
% Se define esta longitud para despues eliminar todas las muestras aÃ±adidas en
% el proceso de operaciones con la seÃ±al.
global longEfectiva;
% si se desea optimizar se pone el nivel al mÃ¡ximo
if(optimizar)
   nivel=4;
end
% cargamos la seÃ±al que nos pasan como parÃ¡metro
senialEnergia=senialEntrada;
% se toma las muestras de la seÃ±al antes del anÃ¡lisis
longEfectiva=length(senialEntrada);
if(strcmp('MÃ¡ximo',nivel))
   nivel=4;
else
   nivel=double(nivel);
end
eleccion=double(eleccionIn);
% se ponen como mÃnimo 640 muestras para poder realizar el anÃ¡lisis
% matricial
senialEnergia=[senialEnergia zeros(1,640-length(senialEnergia))];

%si no ha elegido una de las TW diseÃ±adas matricialmente, se llama a los
%mÃ©todos diseÃ±ado para el anÃ¡lisis con funciones de matlab
if(eleccion<5)
   % se hace el anÃ¡lisis matricial
   senialout=seleccionDeCoeficientes1(senialEnergia,longDep,nivel,eleccion,optimizar);
else
      % se normaliza la seÃ±al
      senialEnergia=normalizacion(senialEnergia')';
      % segÃºn el tipo, se llama a un mÃ©todo o a otro
      load tipoFmatlab;
      analisis=tipoFmatlab(1);
      energiaOff=tipoFmatlab(2);

      %en principio considero que la seÃ±al no esta en energia
      energia=1;
      % si se ha seleccionado la seÃ±al en energÃa se hace el anÃ¡lisis tal
      % cual
```

```
        if(energiaOff)
            load senialAnalizada;
            dire=senialAnalizada;
            senialAux1=load(dire);
            senialAux2=struct2array(senialAux1);
            [fila,columna]=size(senialAux2);
            if(fila>1)
                senial=senialAux2(:,5);
            else
                senial=senialAux2(1,:);
                senial=senial';
            end
            energia=0;
            senialEnergia=senial';
        end
        % seleccionamos el mÃ©todo que correspondiente
        switch analisis
            case 1
                analisisDaubechies(senialEnergia,longDep,nivel,energia,1);
            case 2
                analisisDaubechies(senialEnergia,longDep,nivel,energia,2);
            case 3
                analisisDaubechies(senialEnergia,longDep,nivel,energia,3);
            case 4
                analisisCoiflets(senialEnergia,longDep,nivel,energia);
            case 5
                analisisBior(senialEnergia,longDep,nivel,energia);
        end
    end
end
```

seleccionDeCoeficientes1(senial,lon_se,nivel,eleccion,optimizar)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Author: Jose Maria Lucas Zaragoza
% Date: 22 Augost 2010
```

```
% MÃ©todo encargado la pimera parte del anÃ¡lisis matricial
% Parametros de entrada:
%   - senial: senial depurada y en energia a anÃ¡lizar
%   - lon_se: longitud de la seÃ±al en tiempo (depurada)
%   - nivel: nivel maximo de la seÃ±al a anÃ¡lizar
%   - elecciÃ³n: tipo de analisis que se va a realizar
%   - optimizar: si se ha seleccionado la opcion de optimizaciÃ³n
% Parametros de salida:
%   - coeficientes: coeficientes de la senial analizada
%   - matrizTotal: matriz con la que se sacan los coeficientes
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function
[coeficientes,matrizTotal]=seleccionDeCoeficientes1(senial,lon_se,nivel,eleccion,optimizar)

% selecciÃ³n de los parÃ¡metros segÃºn la elecciÃ³n que se hizo
[alfa,beta,desplazamiento]=seleccionParametros(eleccion);
% normalizado de la seÃ±al
senial=normalizacion(senial')';

%preparaciÃ³n de la matriz
matrizV=zeros(round(length(senial)/2),length(senial));
matrizW=zeros(round(length(senial)/2),length(senial));
matrizV(1,:)=[alfa zeros(1,length(senial)-length(alfa))];
matrizW(1,:)=[beta zeros(1,length(senial)-length(beta))];

% RealizaciÃ³n de la matriz inicial (para el nivel uno)
switch eleccion
    % Haar
    case{1}
        for i=2:(length(senial)/2)
            inicio=zeros(1,(i-1)*desplazamiento);
            matrizV(i,:)=[inicio alfa zeros(1,length(senial)-(length(inicio)+length(alfa)))];
            matrizW(i,:)=[inicio beta zeros(1,length(senial)-(length(inicio)+length(beta)))];
        end
    % Daubechies db2
    case{2}
        %el algoritmo es similar al anterior, solo cambia en la Ãºltima fila

        for i=2:(length(senial)/2)
            if i<length(senial)/2
                inicio=zeros(1,(i-1)*desplazamiento);
                matrizV(i,:)=[inicio alfa zeros(1,length(senial)-(length(inicio)+length(alfa)))];
                matrizW(i,:)=[inicio beta zeros(1,length(senial)-(length(inicio)+length(beta)))];
            else
                mitad=length(alfa)/2;
                matrizV(i,:)=[alfa(mitad+1:end) zeros(1,length(senial)-(length(alfa)))
alfa(1:mitad)];
                matrizW(i,:)=[beta(mitad+1:end) zeros(1,length(senial)-(length(beta)))
beta(1:mitad)];
            end
```

```
    end
  % Daubechies db3
  case{3}
    %el algoritmo es similar al anterior, solo cambia en las dos Ãºltimas filas

    for i=2:(length(senial)/2)
      if i+1<length(senial)/2
        inicio=zeros(1,(i-1)*desplazamiento);
        matrizV(i,:)=[inicio alfa zeros(1,length(senial)-(length(inicio)+length(alfa)))];
        matrizW(i,:)=[inicio beta zeros(1,length(senial)-(length(inicio)+length(beta)))];
      else
        if  i<length(senial)/2
          mitad=length(alfa)/2;
          matrizV(i,:)=[alfa(mitad+2:end) zeros(1,length(senial)-(length(alfa)))
alfa(1:mitad+1)];
          matrizW(i,:)=[beta(mitad+2:end) zeros(1,length(senial)-(length(beta)))
beta(1:mitad+1)];
        else
          mitad=length(alfa)/2;
          matrizV(i,:)=[alfa(mitad:end) zeros(1,length(senial)-(length(alfa))) alfa(1:mitad-
1)];
          matrizW(i,:)=[beta(mitad:end) zeros(1,length(senial)-(length(beta)))
beta(1:mitad-1)];
        end
      end
    end
    % Coiflets
  case{4}
    for i=1:(length(senial)/2)
      if (i==1)
        matrizV(i,:)=[alfa(3:6) zeros(1,length(senial)-(length(alfa))) alfa(1:2)];
        matrizW(i,:)=[beta(3:6) zeros(1,length(senial)-(length(beta))) beta(1:2)];
      else
        if (i==(length(senial)/2))
          matrizV(i,:)=[alfa(5:6) zeros(1,length(senial)-(length(alfa))) alfa(1:4)];
          matrizW(i,:)=[beta(5:6) zeros(1,length(senial)-(length(beta))) beta(1:4)];
        else
          if (i==2)
            matrizV(i,:)=[alfa zeros(1,length(senial)-length(alfa))];
            matrizW(i,:)=[beta zeros(1,length(senial)-length(beta))];
          else
            inicio=zeros(1,(i-2)*desplazamiento);
            matrizV(i,:)=[inicio alfa zeros(1,length(senial)-(length(inicio)+length(alfa)))];
            matrizW(i,:)=[inicio beta zeros(1,length(senial)-
(length(inicio)+length(beta)))];
          end
        end
      end
    end
end

niveltope=nivel;
```

```
matrizWtotal=matrizW;
matrizTotal=[matrizV;matrizW];
porcentage=100*1/(niveltope*2);

% Una vez se ha realizado la primera matriz ahora se pasa a calcular los
% parÃ¡metros del primer nivel y las matrices de los siguientes

for niveles=1:niveltope
   %Si estamos hallando el nivel uno para cualquiera que sea el tipo de
   %transformada (ya hallada en su nivel uno)
   if (niveles==1)
      % se calculan los coeficientes
      aActual=matrizV*senial';
      dActual=matrizW*senial';
      diferencias=dActual;
      coeficientes=[aActual;diferencias];
      % se llama al mÃ©todo de optimizaciÃ³n de los coeficientes

[SenialActual,impulsos,numImpul,optimizado,energiaCoef]=seleccionDeCoeficientes2(coefic
ientes,matrizTotal,eleccion,lon_se,niveles);
      % se calcula la correlaciÃ³n
      correlacion(niveles)=corr2(SenialActual,senial');
      senialFinal(niveles,:)=SenialActual;
      numImpulFinal(niveles)=numImpul;

      %preparaciÃ³n de la presentaciÃ³n en la interfaz
      cadena='Calculada la seÃ±al de nivel 1';
      senialAux=SenialActual;
      nivel=1;


     %EjecuciÃ³n de la interfaz
     %si elegimos la opziÃ³n de optimizar y ademÃ¡s estÃ¡ optimizado bien
     if(optimizar)
        if(optimizado)
           % Llamamos a la interfaz para la representaciÃ³n de los
           % datos, una vez se ha realizado el anÃ¡lisis
           PFCanalisisWavelet(senial,senialAux,
numImpul,correlacion(nivel),coeficientes,lon_se,nivel,impulsos,energiaCoef);
        end
     else
        % Llamamos a la interfaz para la representaciÃ³n de los
        % datos, una vez se ha realizado el anÃ¡lisis
        PFCanalisisWavelet(senial,senialAux,
numImpul,correlacion(nivel),coeficientes,lon_se,nivel);
     end
   else
      %CÃ¡lculo de la matriz del siguiente nivel
      switch eleccion
         % Haar
         case{1}
```

```
            beta=(1/sqrt(2^(niveles-1)))*[alfa -alfa];
            alfa=(1/sqrt(2^(niveles-1)))*[alfa alfa];
            matrizNuevaV=zeros(round(length(senial)/(2^niveles)),length(senial));
            matrizNuevaW=zeros(round(length(senial)/(2^niveles)),length(senial));
            matrizNuevaV(1,:)=[alfa zeros(1,length(senial)-(2^niveles))];
            matrizNuevaW(1,:)=[beta zeros(1,length(senial)-(2^niveles))];
            for i=2:(round(length(senial)/(2^niveles)))
               matrizNuevaV(i,:)=[zeros(1,(i-1)*2^niveles) alfa zeros(1,(length(senial)-(i-
1)*2^niveles-length(alfa)))];
               matrizNuevaW(i,:)=[zeros(1,(i-1)*2^niveles) beta zeros(1,(length(senial)-(i-
1)*2^niveles-length(beta)))];
            end
          % db2
         case{2}
            %Borrado y reajuste de las dimensiones de la matriz
            matrizNuevaV=zeros(length(senial)/(2^niveles),length(senial));
            matrizNuevaW=zeros(length(senial)/(2^niveles),length(senial));
            for j=1:desplazamiento:(length(senial)/(2^(niveles-1)))
               if (((j+2)<(length(senial)/(2^(niveles-1)))))
                     matrizNuevaV(((j-1)/2)+1,:)=alfa(1:end)*matrizV((j:j+(length(alfa)-1)),:);
                     matrizNuevaW(((j-1)/2)+1,:)=beta(1:end)*matrizW((j:j+(length(beta)-1)),:);
               else
                     matrizNuevaV(((j-1)/2)+1,:)=alfa(1:(length(alfa)/2))*matrizV((j:end),:)+...
                        alfa(((length(alfa)/2)+1):end)*matrizV((1:2),:);
                     matrizNuevaW(((j-1)/2)+1,:)=beta(1:(length(beta)/2))*matrizW((j:end),:)+...
                        beta(((length(beta)/2)+1):end)*matrizW((1:2),:);
               end
            end
          % dB3
        case{3}
           %Borrado y reajuste de las dimensiones de la matriz
           matrizNuevaV=zeros(length(senial)/(2^niveles),length(senial));
           matrizNuevaW=zeros(length(senial)/(2^niveles),length(senial));
           for j=1:desplazamiento:(length(senial)/(2^(niveles-1)))
              if(((j+4)<(length(senial)/(2^(niveles-1)))))
                 matrizNuevaV(((j-1)/2)+1,:)=alfa*matrizV(j:(j+length(alfa)-1),:);
                 matrizNuevaW(((j-1)/2)+1,:)=beta*matrizW(j:(j+length(alfa)-1),:);
              else
                 if (((j+2)<(length(senial)/(2^(niveles-1)))))
                    matrizNuevaV(((j-1)/2)+1,:)=alfa(5:6)*matrizV((1:2),:)+...
                       alfa(1:((length(beta)/2)+1))*matrizV((j:end),:);
                    matrizNuevaW(((j-1)/2)+1,:)=beta(5:6)*matrizW((1:2),:)+...
                       beta(1:((length(beta)/2)+1))*matrizW((j:end),:);
                 else
                    matrizNuevaV(((j-
1)/2)+1,:)=alfa(((length(alfa)/2)):end)*matrizV((1:(length(alfa)/2)+1),:)+...
                       alfa(1:((length(alfa)/2)-1))*matrizV((j:end),:);
                    matrizNuevaW(((j-
1)/2)+1,:)=beta(((length(beta)/2)):end)*matrizW((1:(length(alfa)/2)+1),:)+...
                       beta(1:((length(beta)/2)-1))*matrizW((j:end),:);
                 end
              end
```

```
            end
            % Coiflets
          case{4}
            %Borrado y reajuste de las dimensiones de la matriz
            matrizNuevaV=zeros(length(senial)/(2^niveles),length(senial));
            matrizNuevaW=zeros(length(senial)/(2^niveles),length(senial));
            for j=1:desplazamiento:(length(senial)/(2^(niveles-1)))
              if(j==1)
                matrizNuevaV(((j-1)/2)+1,:)=alfa(1:(length(alfa)/2)-
1)*matrizV(((length(senial)/2^(niveles-1))-j:(length(senial)/2^(niveles-1))),:)+...
                  alfa((length(alfa)/2):end)*matrizV(1:(length(alfa)/2)+1,:);
                matrizNuevaW(((j-1)/2)+1,:)=alfa(1:(length(alfa)/2)-
1)*matrizW(((length(senial)/2^(niveles-1))-j:(length(senial)/2^(niveles-1))),:)+...
                  beta((length(alfa)/2):end)*matrizW(1:(length(alfa)/2)+1,:);
              else
                if((j+((length(alfa)/2)+1))<(length(senial)/(2^(niveles-1))))
                  matrizNuevaV(((j-1)/2)+1,:)=alfa*matrizV(j:(j+length(alfa)-1),:);
                  matrizNuevaW(((j-1)/2)+1,:)=beta*matrizW(j:(j+length(alfa)-1),:);
                else
                  matrizNuevaV(length(senial)/(2^(niveles-
1)),:)=alfa(1:(length(alfa)/2)+1)*matrizV((1:(length(alfa)/2)+1),:)+...
                   alfa((length(alfa)/2)+2:end)*matrizV(((length(senial)/(2^(niveles-1)))-
1):(length(senial)/(2^(niveles-1))),:);
                    matrizNuevaW(length(senial)/(2^(niveles-
1)),:)=beta(1:(length(alfa)/2)+1)*matrizW((1:(length(alfa)/2)+1),:)+...
                  beta((length(alfa)/2)+2:end)*matrizW(((length(senial)/(2^(niveles-1)))-
1):(length(senial)/(2^(niveles-1))),:);
                end
              end
            end

      end

      matrizV=matrizNuevaV;
      matrizW=matrizNuevaW;
      matrizWtotal=[matrizW;matrizWtotal];
      matrizCompuesta=[matrizV;matrizWtotal];
      matrizTotal=matrizCompuesta;
      disp(['paso siguiente. Hallada matriz de nivel ' int2str(niveles)]);

      % esto es comÃºn a todos los tipos

      aActual=matrizV*senial';
      dActual=matrizW*senial';
      diferencias=[dActual;diferencias];
      coeficientes=[aActual;diferencias];


[SenialActual,impulsos,numImpul,optimizado,energiaCoef]=seleccionDeCoeficientes2(coefic
ientes,matrizTotal,eleccion,lon_se,niveles);
      correlacion(niveles)=corr2(SenialActual,senial');
      senialFinal(niveles,:)=SenialActual;
```

```
        numImpulFinal(niveles)=numImpul;

        switch niveles
           case{2}
              cadena='Calculada la seÃ±al de nivel 2';
              senialAux=SenialActual;
              nivel=2;

              % ejecutamos nuestra interfaz de nuevo para presentar los
              % datos analizados
              if(optimizar)
                  if(optimizado)

PFCanalisisWavelet(senial,SenialActual,numImpul,correlacion(nivel),coeficientes,lon_se,niv
eles,impulsos,energiaCoef);
                  end
              else

PFCanalisisWavelet(senial,SenialActual,numImpul,correlacion(nivel),coeficientes,lon_se,niv
eles);
              end
              case{3}
              cadena='Calculada la seÃ±al de nivel 3';

              senialAux=SenialActual;
              nivel=3;

              % ejecutamos nuestra interfaz de nuevo para presentar los
              % datos analizados
             if(optimizar)
                  if(optimizado)

PFCanalisisWavelet(senial,SenialActual,numImpul,correlacion(nivel),coeficientes,lon_se,niv
eles,impulsos,energiaCoef);
                  end
                  %hay que fijarse en que no se llama a la interfaz si
                  %no se ha optimizado
             else
                 PFCanalisisWavelet(senial,SenialActual,
numImpul,correlacion(nivel),coeficientes,lon_se,niveles);
             end
             case{4}
              cadena='Calculada la seÃ±al de nivel 4';

              senialAux=SenialActual;
              nivel=4;

              % ejecutamos nuestra interfaz de nuevo para presentar los
              % datos analizados
              if(optimizar)
                 if(optimizado)
```

```
PFCanalisisWavelet(senial,SenialActual,numImpul,correlacion(nivel),coeficientes,lon_se,niv
eles,impulsos,energiaCoef);
                end
            else


PFCanalisisWavelet(senial,SenialActual,numImpul,correlacion(nivel),coeficientes,lon_se,niv
eles);
            end
      end
   end
end
```

seleccionDeCoeficientes2(coeficientes,matrizTotal,tipo,lon_se,nivel)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Author: JosÃ© MarÃa Lucas Zaragoza
% Date: 22 Augost 2010
% Este mÃ©todo se encarga, de una vez hallado los coeficientes, hacer la
% seleccion de forma que se escojan los coeficientes idoneos para hacer la
% seleccion
% Parametros de entrada:
%   - coeficientes: Coeficientes calculados en la primera seleccion
%   - matrizTotal: Matriz de calculo de los coeficientes
%   - tipo: tipo de transformada de wavelet
%   - lon_se: longitud de la seÃ±al depurada en tiempo
%   - nivel: nivel maximo de analisis
% Parametros de salida:
%   - SenialActual: senial que ha sido analizada
%   - impulsos: impulsos detectados
%   - numImpul: numero de impulsos detectado
%   - optimizado: si se ha optado por el metodo de optimizado
%   - coeficientes_energia: energia de los impulsos
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


function
[SenialActual,impulsos,numImpul,optimizado,coeficientes_energia]=seleccionDeCoeficiente
s2(coeficientes,matrizTotal,tipo,lon_se,nivel)
    %deteccian es un booleano que dira si se ha llegado a un
    %resultado concluyente y si se puede hallar los coeficientes en
    %energÃa.
    deteccion=0;
    coeficientes_energia=0;
```

```
        impulsos=0;
        optimizado=0;
        % se halla la senial con los coeficientes de aproximacion
        % correspondientes al nivel que se pase
        SenialActual=matrizTotal(1:(640/2^nivel),:)'*coeficientes(1:(640/2^nivel));
        SenialActual=normalizacion(SenialActual);
        %se comprueban si existen 6 impulsos
        [impulsos,numImpul]=deteccionImpulsosEnergia(SenialActual,lon_se);
        %si detecta de manera correcta, estos seran los Ãºltimos coeficientes
        %seleccionados
        if(numImpul==6)
           optimizado=1;
           disp(['deteccion correcta --> Coeficientes seleccionados: nivel ' int2str(nivel) ' paso 1']
);
           coeficientesFinales=coeficientes(1:(640/2^nivel));
           disp(['coeficientes de aproximacion: ' int2str(size(coeficientesFinales))]);
           numeroCoeficientes=length(coeficientesFinales(:,1));
           save numeroCoeficientes;
           deteccion=1;
        %En caso contrario
        else
           %habra que seleccionar la seÃ±al en diferencias que
           %contiene informaciÃ³n que se necesita

coeficientesActuales=[coeficientes(1:(640/2^nivel));(coeficientes((640/2^nivel)+1:(640/2^niv
el)*2))];
           SenialActual=matrizTotal(1:length(coeficientesActuales),:)'*coeficientesActuales;
           SenialActual=normalizacion(SenialActual);
           %se vuelve a comprobar si se detectan los 6 impulsos
           [impulsos,numImpul]=deteccionImpulsosEnergia(SenialActual,lon_se);
           if(numImpul==6)
              optimizado=1;
              disp(['deteccion correcta --> Coeficientes seleccionados: nivel ' int2str(nivel) ' paso
2'] );
              coeficientesFinales=coeficientesActuales;
              disp(['coeficientes de aproximacion y difenrecias: '
int2str(size(coeficientesFinales))])
              deteccion=1;
           %en caso de que tampoco sea suficiente con los coeficientes en
           %direrencias de mi nivel, se cogen tb los del anterior
           else
              if(nivel>2)
              coeficientesActuales=[coeficientesActuales;
coeficientes((length(coeficientesActuales)+1):((length(coeficientesActuales)+1)+(640/2^(niv
el-1))))];
              SenialActual=matrizTotal(1:length(coeficientesActuales),:)'*coeficientesActuales;
              SenialActual=normalizacion(SenialActual);
              %se realiza de nuevo el mismo procedimiento
              [impulsos,numImpul]=deteccionImpulsosEnergia(SenialActual,lon_se);
              end
              if(numImpul==6)
                 optimizado=1;
```

```
                disp(['deteccion correcta --> Coeficientes seleccionados: nivel ' int2str(nivel) '
paso 3' ] );
                coeficientesFinales=coeficientesActuales;
                disp(['coeficientes de aproximacion y difenrecias nivel anterior: '
int2str(size(coeficientesFinales))])
                deteccion=1;
             end
          end

       end
       %si se ha encontrado los coeficientes se hallan los coeficientes
       %en energÃa y se devuelven
       if deteccion
          coeficientes_energia=energyCoeficient(SenialActual,impulsos,numImpul,lon_se);
       end
```

seleccionParametros(eleccion)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Author: JosÃ© MarÃa Lucas Zaragoza
% Date: 27 Augost 2010
% Metodo encargado la seleccion de los parametros necesarior para la
% composicion de la matriz usada para la deteccion de los coeficientes
% Parametro de entrada:
%   - eleccion: tipo de transformada de wavelet
% Parametros de salida:
%   - alfa,beta,desplazamiento: parametros para la construccion de la
%   matriz
%   - eleccion: tipo de transformada seleccionada
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [alfa,beta,desplazamiento,eleccion]=seleccionParametros(eleccion)
%inicializacion de las variables a usar
haar=0;
db2=0;
db3=0;
coiflets=0;
%dendiendo del metodo se cogen unos parametros u otros
switch eleccion
   %seleccion de parametros
   %haar
   case{1}
      haar=1;
      alfa=[1/sqrt(2) 1/sqrt(2)];
      beta=[1/sqrt(2) -1/sqrt(2)];
      desplazamiento=2;
   %Wavelet Daubechies db2
```

```
    case{2}
        db2=1;
        alfa(1)=[(1+sqrt(3))/4*sqrt(2)];
        alfa(2)=[(3+sqrt(3))/4*sqrt(2)];
        alfa(3)=[(3-sqrt(3))/4*sqrt(2)];
        alfa(4)=[(1-sqrt(3))/4*sqrt(2)];
        beta(1)=alfa(4);
        beta(2)=-alfa(3);
        beta(3)=alfa(2);
        beta(4)=-alfa(1);
        desplazamiento=2;
    %Wavelet Daubechies db3
    case{3}
        db3=1;

        alfa(1)=0.332670552950083;
        alfa(2)=0.806891509311092;
        alfa(3)=0.459877502118491;
        alfa(4)=-0.135011020010255;
        alfa(5)=-0.0854412738820267;
        alfa(6)=0.0352262918857095;
        beta(1)=alfa(6);
        beta(2)=-alfa(5);
        beta(3)=alfa(4);
        beta(4)=-alfa(3);
        beta(5)=alfa(2);
        beta(6)=-alfa(1);
        desplazamiento=2;
    %Coiflets
    case{4}
        coiflets=1;
        alfa(1)=(1-sqrt(7))/16*sqrt(2);
        alfa(2)=(5+sqrt(7))/16*sqrt(2);
        alfa(3)=(14+2*sqrt(7))/16*sqrt(2);
        alfa(4)=(14-2*sqrt(7))/16*sqrt(2);
        alfa(5)=(1-sqrt(7))/16*sqrt(2);
        alfa(6)=(-3*sqrt(7))/16*sqrt(2);
        beta(1)=alfa(6);
        beta(2)=-alfa(5);
        beta(3)=alfa(4);
        beta(4)=-alfa(3);
        beta(5)=alfa(2);
        beta(6)=-alfa(1);
        desplazamiento=2;
end
```